An Architectural Framework for Assessing Quality of Experience of Web Applications

A thesis submitted for the degree of Doctor of Philosophy

By

Omar Amer Radwan

Department of Computer Science

Brunel University London

March 2017

ABSTRACT

Web-based service providers have long been required to deliver high quality services in accordance with standards and customer requirements. Increasingly, however, providers are required to think beyond service quality and develop a deeper understanding of their customers' Quality of Experience (QoE). Whilst models exist that assess the QoE of Web Application, significant challenges remain in defining QoE factors from a Web engineering perspective, as well as mapping between so called 'objective' and 'subjective' factors of relevance. Specifically, the following challenges are considered as general fundamental problems for assessing QoE: (1) Quantifying the relationship between QoE factors; (2) predicting QoE as well as dealing with the limited data available in relation to subjective factors; (3) optimising and controlling QoE; and (4) perceiving QoE. In response, this research presents a novel model, called QoEWA (and associated software instantiation) that integrates factors through Key Performance Indicators (KPIs) and Key Quality Indicators (KQIs). The mapping is incorporated into a correlation model that assesses QoE, in particular, that of Web Application, with a consideration of defining the factors in terms of quality requirements derived from web architecture. The data resulting from the mapping is used as input for the proposed model to develop artefacts that: quantify, predict, optimise and perceive QoE. The development of QoEWA is framed and guided by Design Science Research (DSR) approach, with the purpose of enabling providers to make more informed decisions regarding QoE and/or to optimise resources accordingly. The evaluation of the designed artefacts is based on a build-and-evaluate cycle that provides feedback and a better understanding of the utilised solutions. The key artefacts are developed and evaluated through four iterations: Iteration 1 utilises the Actual-Versus-Target approach to quantify QoE, and applies statistical analysis to evaluate the outputs. Iteration 2: utilises a Machine Learning (ML) approach to predict QoE, and applies statistical tests to compare the performance of ML algorithms. Iteration 3 utilises the Multi-Objective Optimisation (MOO) approach to optimise QoE and control the balance between resources and user experience. Iteration 4 utilises the Agent-Based Modelling approach to perceive and gain insights into QoE. The design of iteration 4 is rigorously tested using verified and validated models.

DEDICATIONS

I dedicate this work to God almighty for giving me an opportunity to learn, gain, and spread knowledge. I also dedicate this work to my parents, for making me who I am today. Truly, your love and support gave me the strength to complete this journey. It is with great pleasure that I dedicate this work to my wife, Dalya, for her love and to my precious son and daughter, Amer and Taleen. I would like to express my gratitude to my sisters and brother for their encouragements and inspirations.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my appreciation and sincere gratitude to my principle supervisor, Professor Mark Lycett, for his invaluable feedback and advice on my thesis. Special thanks go to Dr. Sergio de Cesare, my second supervisor, for his continuous valuable support, advice and encouragement.

A special thank you also goes to all my colleagues in the Information Services Department, in particular my manager Mr Iain Liddell for his kind support as well as his giving me a flexible work arrangement that has enabled me to complete this work. Thanks are extended to my colleagues on Project TIGER for their support.

I dedicate this thesis to my beloved parents and my wonderful wife as well as my kids for their endless love and continues support and encouragement. This work would not be possible without the encouragement of my mother in-law, sisters and brother. A big thanks go to every member of my family as well as friends for supporting me spiritually throughout my PhD study.

I am very grateful to have the opportunity to work and study at Brunel University, London. Many thanks to everyone at the University, in particular, the department of Computer Science for giving me the environment and the excellent resources to undertake this work.

DECLARATION

The following papers have been submitted as a result of the research conducted in this PhD thesis:

O. Radwan and M. Lycett, 2017. Developing a Quality of Experience (QoE) Model for Web Applications. Submitted to Information Systems journal (ISJ) (Under Revision).

O. Radwan, 2017. An Architectural Model for Managing Quality of Experience (QoE) of Web Services. Submitted to The Ninth International Conference on Ubiquitous and Future Networks. Milan, Italy: IEEE (Accepted).

ABBREVIATIONS

ABM	Agent-Based Modelling
CCI	Correct Classified Instances
DSR	Design Science Research
DT	Decision Tree
FP	False Positive
GA	Genetic Algorithm
IS	Information System
ISO	International Organisation for Standardisation
ITU	International Telecommunication Union
KNN	K-Nearest Neighbours
KPI	Key Performance Indicator
KQI	Key Quality Indicator
MAE	Mean Absolute Error rate
ML	Machine Learning
MOO	Multi-Objective Optimisation
MOS	Mean Opinion Score
NB	Naive Bayes
NPS	Net Promotor Score
OASIS	Organisation for the Advancement of Structured Information Standards
ODD	Overview, Design concept, and Details
QoE	Quality of Experience
QoEWA	Quality of Experience of Web Application
QoS	Quality of Service
RF	Random Forest
ROC	Receiver Operating Characteristic model
SLA	Service Level Agreement
SMO	Sequential Minimal Optimisation
SOA	Service-Oriented Architecture
TOGAF	The Open Group Architecture Framework
TP	True Positive
UML	Unified Modelling Language
WEKA	Waikato Environment for Knowledge Analysis
WIS	Web-based Information Systems
WOM	Word of Mouth

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

1.1 Overview

This chapter presents an overview of the research and outlines the scope and the context of this thesis as well as providing an introduction to QoE in the Web Application context. Section 1.2 presents the research background and motivation and briefly describes the problem and the suggested solution. Section 1.3 presents the research question, along with aim and objectives, whilst Section 1.4 describes the research approach adopted in this thesis. Section 1.5 presents the overall structure of this thesis.

1.2 Background and Research Motivations

In recent literature, it is generally agreed that Web Application is no longer simple and has become increasingly sophisticated, involving complex interactions between several technologies (Lew et al. 2012; Tupamäki & Mikkonen 2013; Das et al. 2016). Web Application technology has evolved towards Cloud technology and service-oriented applications that interact dynamically over the internet with intelligent software applications (Chieu et al. 2009). With this evolution, the demand for high quality of Web Application has increased considerably and it has become an important aspect in web engineering (Alhamazani et al. 2015).

As a result, many studies have been carried out from different disciplinary perspectives to assess and improve the quality of Web Application. Most of these studies presented quality models, based on standards such as ISO-9126, ISO-25010, ISO-9241, TOGAF and OASIS (Behkamal et al. 2009; Bevan et al. 2015; OASIS 2006), or Quality of Service (QoS) models, based on Service-oriented Architecture (SOA) formation, such as architectural models (Ran 2003; O’Brien et al. 2007;

Zeng et al. 2007; Hasan et al. 2012; Alhamazani et al. 2015). These models generally refer to the objective aspects of quality, e.g. performance and reliability, neglecting that it is pointless to improve the quality objectively, without considering the subjective aspects of quality, which are generally related to customer requirements. Hence, these models fail to understand the relationship between what is offered by the service provider and what is expected by the users (Offutt 2002b).

In contrast, other studies have gone beyond the objective aspects of quality and have involved developing a deeper understanding of the subjective aspects to improve the quality of service and customer satisfaction as well as correlating the relationship between the objective and subjective aspects (Offutt 2002a; Cecchet et al. 2013; Phillips et al. 2015). Consequently, Quality of Experience (QoE), which has been traditionally used for multimedia services (Baraković & Skorin-Kapov 2013), has been recently adopted for web quality to formulate the relationship between the objective and subjective aspects (Nguyen et al. 2013; ITU-T 2014; Skorin-kapov & Barakovic 2015; Yamauchi et al. 2015). QoE is considered as a multidisciplinary concept that interacts with human, information, and technology (Laghari & Connelly 2012; Geerts et al. 2010). Its measurement is usually performed by a combination of so-called factors, which are termed 'objective' and 'subjective' (Mitra et al. 2011). The former are typically measured by QoS parameters, such as web page latency, bandwidth, or delay (Brooks & Hestnes 2010), while the latter are typically measured by Mean Opinion Score (MOS) tests, which assess how service quality is perceived by customers (Khan et al. 2012).

In the context of Web Application, QoE has been utilised for Cloud applications (Hobfeld et al. 2012) as well as for multimedia web services and applications (Nguyen et al. 2013; ITU-T 2014; Skorin-kapov & Barakovic 2015; Yamauchi et al. 2015). Furthermore, other studies have drawn attention to loosely-coupled and interoperable services, such as Zieliński et al. (2012), who proposed an adaptive SOA solution that integrates SOA with QoS and QoE. In terms of web service discovery and composition, QoE was adopted to assess user perception about the quality of Web services (Upadhyaya et al. 2014; Upadhyaya et al. 2015).

However, it is noted that most QoE studies have not paid attention to the aspect of Information Systems (IS), particularly Web-based Information Systems (WIS) , which provide the knowledgebase and methodologies (Hevner et al. 2004; Cassidy & Hamilton 2016) required for developing conceptual and technological artefacts for Web Application (Oztekin et al. 2009). Moreover, there is still a lack of rigour in defining the QoE factors as most of current QoE models are based on ITU factors (ITU-T 2006b; ITU-T 2014), which are extracted from network and multimedia domains, rather than Web Application ones. Consequently, they fail to define the QoE factors extracted from web quality requirements and web architecture design, i.e. the absence of formulating the web QoE factors from the core of web application architecture forms naive factors and metrics that may not meet web and software quality requirements, which are generally derived from the standards (ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2012).

Alongside the above challenges, which are particularly related to QoE of Web Application, it is found in the literature that the following challenges are considered as general fundamental problems for the process of QoE assessment in most application domains (Skorin-kapov 2012; Hobfeld et al. 2012; Baraković & Skorin-Kapov 2013).

- **The quantification of QoE**: There is still a general problem in quantifying QoE in the traditional QoE models (Alreshoodi & Woods 2013; Aroussi & Mellouk 2014). This actually refers to the lack of the mechanisms that define the QoE factors and their relationships (Fiedler et al. 2010; Schatz et al. 2013; Laghari & Connelly 2012), as well as the methods used for scaling, measuring, prioritising and weighting the QoE factors (Van et al. 2008; Zinner et al. 2010).

- **The prediction of QoE:** Due to the complex nature of the MOS process, which requires continues feedback from end users regarding their satisfaction with the provided service (Elkotob et al. 2010), QoE prediction becomes important in QoE assessment of foreseeing the level of perceived QoE

(Menkovski, Liotta, et al. 2009). However, there is still a significant challenge in emerging and understanding the correlation between the objective and subjective factors and their influences so as to improve the prediction of QoE (Mushtaq et al. 2012; Aroussi & Mellouk 2014).

- **The optimisation of QoE:** Since the nature of QoE varies from business to business and from technology to technology (Al-Moayed & Hollunder 2010), there is still a challenge in emerging and understanding the characteristics of QoE factors and how they are associated to the resources to be optimised (Sharma et al. 2012; Song et al. 2012). Moreover, there is still ongoing debate concerning the technique used for evaluating and adjusting the balance between the objective and subjective factors of QoE (Baraković & Skorin-Kapov 2013).

- **The user perception of QoE**: QoE does not always reflect the user’s levels of satisfaction as he/she may be affected by other external factors, such as social network or Word of Mouth (WOM) communications (Hummel et al. 2012). It is still a challenge to answer the question: Why do some users rate poor service as excellent in contrast to others who rate excellent service as bad? (Dusi et al. 2012). It is important to evaluate user satisfaction and brand loyalty in QoE optimisation processes as they are prohibitively expensive and can have detrimental consequences (Nokia 2004; Soldani et al. 2006).

With the above challenges in mind, for this research a novel model appropriate for assessing the QoE of Web Applications is proposed (called QoEWA from this point) that integrates the Key Performance Indicators (KPIs) and Key Quality Indicators (KQIs). The KPIs encompass all the objective factors, while the KQIs encompass all the subjective factors. The mapping between the KPIs and KQIs is incorporated into a correlation model that assesses the QoE of Web Applications, with a consideration of defining the factors in term of quality requirements derived from web architecture, i.e. the defined factors are based on standard models, such as (ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS

2012), and usability models such as (Seffah et al. 2006; Mifsud 2015; Hussain & Kutar 2009). The mapping consists of three processes: (1) The KPIs assessment process; (2) the KQIs assessment process; and (3) the mapping process. The data resulting from the mapping are used as the input of the proposed model to assess QoE.

Following Design Science Research (DSR) as an IS approach, QoEWA is iteratively developed to provide artefacts that quantify, predict, optimise, and perceive QoE. Through this research two different types of artefacts have been developed, including the QoEWA (conceptual) model and its (technological) instantiation, each of which is created in two complementary phases: (1) A behavioural science phase that carries out the research through the development and justification of theories; (2) and a design science phase that carries out the research through the building and evaluation of artefacts. This allows for the incremental development of QoEWA through a build-evaluate cycle, starting from the artefacts which facilitates the relationship between KPIs and KQIs and ending up eventually with IS artefacts incorporated into QoEWA as utilities, including:

- Artefacts for quantifying QoE: A set of artefacts are developed in this research to enable researchers and practitioners to gain theoretical and technical knowledge that connects QoE measurement theories (e.g. Alreshoodi & Woods 2013; Aroussi & Mellouk 2014) to the gap analysis technique for facilitating the QoE measurement by quantifying the relationship between the KPIs and KQIs.

- Artefacts for predicting QoE: A set of artefacts are developed to enable researches and practitioners to gain theoretical knowledge and practical experience in the fundamental aspects of designing, building, and applying ML approach in the area of QoE of Web Application i.e. this allows QoE assessor to understand the links and requirements that bridge between the service quality and user experience.

- Artefacts for optimising QoE: A set of artefacts are developed to allow researchers and practitioners to gain knowledge that links between QoE optimisation (e.g. Baraković & Skorin-Kapov 2013) and MOO technique to practically optimise quality and control the balance between resources and user experience.

- Artefacts for Perceiving QoE: A set of artefacts are developed with a link to the artefacts developed for predicting QoE to enable the ability to perceive user's QoE. An ABM is proposed as an analytical technique for QoE of Web application to enable researchers and practitioners to obtain knowledge and practical experience of developing and applying an ABM in the area of QoE.

1.3 Research Aim and Objectives

This research project is aimed at facilitating the objective and subjective assessments of Web Application using a QoE approach, as well as answering the following research question: How can the Key Performance Indicators (KPIs) and the Key Quality Indicators (KQIs) be identified and mapped to assess the Quality of Experience (QoE) of Web Application?

To fulfil the aim which is embodied in the research question presented above, four research objectives are set as follows:

- **Objective 1:** To present a review of the state of the art in QoE of Web Application and associated disciplines to elicit understanding of the challenges related to the QoE assessments process;

- **Objective 2:** To conceptualise a solution based on the correlation between the KPIs and KQIs that facilitates the assessment of QoE of Web Application with consideration about the aspects related to web quality requirements and web architecture;

- **Objective 3:** To design and develop theoretical and technical artefacts that have the capability of mapping the KPIs and KQIs as well as quantifying, predicting, optimising, and perceiving QoE of Web Application;
- **Objective 4:** To demonstrate the features of the developed model (QoEWA) and evaluate its capabilities and limitations in the context of Web Application and DSR.

1.4 Research Approach

Since this research aims to contribute to the knowledge base of the QoE of Web Application from a wider Information System (IS) perspective on problem-solving environment, various research approaches can be followed such as action learning, concept mapping, action research, and Design Science Research (DSR) to develop IS artefacts, including conceptual and technological artefacts. However, the multidisciplinary nature of QoE (Laghari & Connelly 2012; Geerts et al. 2010) requires a research approach such as DSR (Cassidy & Hamilton 2016), to study the nature of the complementarity between the behavioural science and design science aspects (Hevner & Chatterjee 2010). Therefore, DSR approach is employed in this research as a general methodological framework. The context of the current research is one where behavioural science is employed to build theories and artefacts that describe end user requirements in relation to QoE. In addition, design science is engaged with to provide the techniques needed for justifying and evaluating QoE models. Taken together, these approaches will help to fulfil the aims of quantifying, predicting, optimising and perceiving QoE.

The basic principle behind DSR is to develop knowledge and understanding of a problem domain in order to build IS artefacts, for which practitioners and professionals have the guidelines to improve their development outcomes and researchers have the opportunity to test hypotheses (Walls et al. 1992). An IS artefact is the outcome of a DSR process, which can be guided and structured by IS design theory (Prat et al. 2014; Jones et al. 2007). The DSR process is generally structured into phases that identify the problem and develop its solution over iterative build-and-evaluate cycles (Vaishnavi, V. and Kuechler 2004; Hevner & Gregor 2013). Specifically, the design process is not linear and work in the solution space often reframes the problem space. If so: (a) Design theory is more ‘grounded’ in practice in a way that should be acknowledged; and (b) iterative and/or incremental learning forms an important part of that theory.

Consequently, the practical work achieved in this research is described in four design-build-evaluate iterations, theoretically, following the popular process models and guidelines of DSR (Hevner et al. 2004; Kuechler & Vaishnavi 2008; Peffers et al. 2008). Moreover, the development of the iterations is achieved through five phases processes (Vaishnavi, V. and Kuechler 2004), which include: (1) Awareness of the problem; (2) solutions selection and suggestion; (3) development; (4) evaluation; and (5) conclusion. In governing the phases and emphasising certain aspects of developing an artefact in a subsequent build cycle, this research involves mapping the phases of Vaishnavi, V. and Kuechler (2004) with the skeleton of a design theory that covers (Jones et al. 2007): (a) The purpose and scope of the theory; (b) constructs; (c) the principles of form and function; (c) artefact mutability; (d) testable propositions; (e) justificatory knowledge (kernel theory); (f) principles of implementation; and (g) each expository instantiation.

Based on the above DSR guidelines and process, the objectives of this research will be achieved and accordingly, the proposed model QoEWA iteratively developed to quantify, predict, optimise and perceive the QoE of Web Application. Figure 1-1 illustrates how the proposed model is framed by the DSR process (Vaishnavi, V. and Kuechler 2004) and guided by the design theory proposed by Jones & Gregor (2007). The output of each iteration addresses three aspects of contributions embodied by Hevner et al. (2004): (1) Contributions to the knowledge base; (2) contributions to the IS research (artefacts); and (3) contributions to the environment.

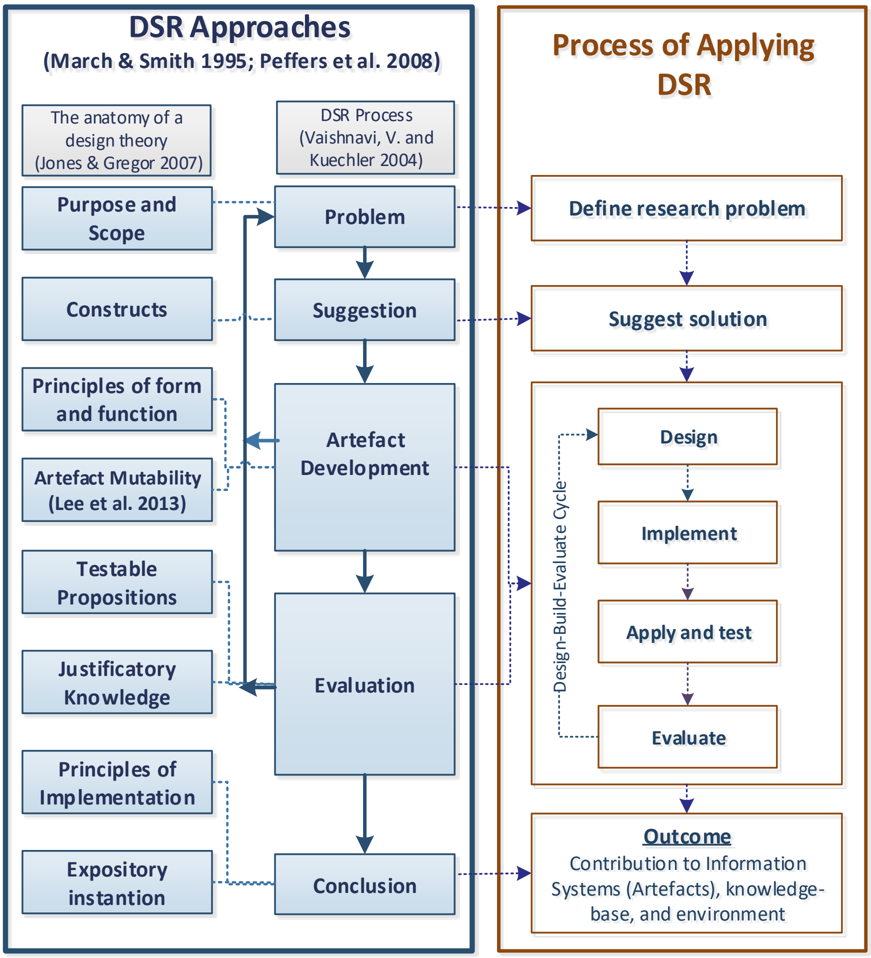


Figure 1-1: Research approach and processes for this research

1.5 Thesis Structure

In accordance with the above, the structure of the thesis is as follows.

Chapter 2: Presents a literature review addressing the problem described in Chapter 1. This is divided into three sections: first, it begins with a brief overview of Web Application and thereafter discusses the importance of understanding web architecture prior to undertaking web quality assessment. This provides a knowledge base for examining the existing web architecture models (e.g. multi-tier, SOA, and MVC) as well as web quality models (e.g. models derived from ISO-9126, ISO -9241, and OASIS). Second, to investigate quality of Web Application from subjective perspectives, the existing QoE models and frameworks are reviewed in terms of identifying their limitations, as well illustrating the journey of QoE, which includes QoE management, modelling, monitoring, measuring, optimising, and controlling processes. Third, the chapter discusses QoE in the context of Web Application exploring the challenges in adopting QoE for web-based services. It discusses the fundamental issues in relation to identifying objective and subjective QoE factors and their relationships.

Chapter 3: Presents an overview of IS research methodologies focusing on DSR methodology, which is adopted in this research. DSR is discussed in terms of informing the design, as a means of expressing design knowledge and as an outcome of the design instantiation. The DSR process is presented with consideration of design theory and the development and evaluation of the artefact. This chapter also explains how the DSR methodology is utilised, addressing the research problem and its suggested solutions. Furthermore, it clarifies how the key artefacts are developed and evaluated through four iterations. The initial iteration is determined from the research problem defined in Chapter 1, whereas, the next three (Iteration 2, 3, and 4) are determined from the knowledge base from Chapter 2 and the evaluation conducted at the end of each iteration. To link the defined problems and the suggested solutions, this chapter provides a conceptual view of the development of QoEWA that describes the whole process of assessing QoE, including the sub-processes that quantify, predict, optimise and perceive QoE. This aims to give the reader a coherent view of the techniques and approaches utilised in each iteration.

Chapter 4: Presents details of the initial iteration of the QoEWA model, describing the conceptual and technological artefacts that are constructed for quantifying QoE. It presents an overview of the performance management approach (Actual-Versus-Target), which is utilised as a technique that examines the relation between the KPIs and KQIs, as well as quantifying QoE. The design of QoEWA develops key components that allow QoE assessors to define QoE factors systematically, in accordance to Web Application architecture. This chapter provides details of the instantiation of Iteration 1, specifying the main software components of QoEWA, which measure the objective and subjective factors and determine the relation between the KPI and KQI values. Moreover, this chapter illustrates the demonstration of QoEWA through empirical tests that: (1) examine the correlation between KPIs and KQIs and (2) quantify QoE. Finally, the results obtained from the tests are analysed and evaluated to examine the efficiency of features provided by the QoEWA model.

Chapter 5: Presents the details of the development of Iteration 2, which extends the design of the QoEWA model to classify and predict, intelligently, the subjective measurements (KQI values). This chapter starts with an overview of the ML approach presenting five supervised learning algorithms derived from the literature review for comparison: (DT, NB, SMO, IBK, and RF). To compare the performance and accuracy of these algorithms, five statistical methods are considered in Iteration 2, including: True Positive (TP), False Positive (FP), Precision, Recall, and the F-measure. Alongside these tests, the Receiver Operating Characteristic model (ROC) approach is used to determine visually the performance. In implementation terms, the Waikato Environment for Knowledge Analysis (WEKA) tool is utilised in this chapter to implement the chosen ML algorithms via WEKA explorer and WEKA knowledge workflow. The results obtained from the prediction process are evaluated and analysed to select the best ML algorithm that can be used for predicting KQI values.

Chapter 6: Presents the details of the development of Iteration 3, which extends the design of the QoEWA model to optimise QoE. This chapter starts with an introduction to the MOO approach for determining the optimal QoE value. A design decision was taken in this iteration to control the balance between resources and user experience. The existing design of QoEWA is extended by incorporating the MOO approach to examine the trade-off between multiple objectives (KPIs and KQIs). The design is implemented in a simulation system that generates a set of points to determine the non-dominated Pareto optimal solutions, which in fact demonstrates the optimal trade-off between KPIs and KQIs. The instantiation of Iteration 3 is implemented as an extension of QoEWA and applied in the context of

Web Application, as performed in Iteration 1 and Iteration 2. Two tests are performed: First, to validate the model by exploring the data and comparing it with the analysis obtained in iteration 1. Second, to compute and predict the set of optimal values.

Chapter 7: Presents details of the development of Iteration 4, which extends the design of the QoEWA model to perceive of and gain insights into QoE. This chapter starts with an introduction to the ABM approach, which is incorporated into QoEWA to assess user-perceived QoE of Web Application when resources are constrained (minimum, maximum or optimum). This allows for better insight and decision-making towards QoE and user satisfaction as well as the evaluation of KPIs and KQIs. The design of the ABM is described by the ODD protocol, which includes three main components (Overview, Design concept, and Details) for developing the agent and its behaviour. The agent represents the end-user and interacts with the environment and other agents. Each agent has properties based on the user's profile, and a behaviour based on the constraints and rules (presented in Iteration 2 as ML rules). The agent is classified into three types (Promoters, Detractors and Passives) using the NPS approach. The ABM system is implemented and tested in three scenarios. The first, pertains to the process that initiates and explores the current situation, whilst the second presents a what-if scenario and examines the agents' behaviour. The third scenario pertains to examining agents' behaviour and interaction.

Chapter 8: Presents an overall summary of this research. There is discussion on how aims and objectives are met by the development of the four iterations that build up QoEWA model. This chapter also considers the main contribution of this research, which provides an architectural framework framed by DSR to assess the QoE of Web Application, along with three aspects building on the work of Hevner et al. (2004), covering: (1) Contributions to the knowledge base; (2) contributions to IS research (artefacts); and (3) contributions to the environment.

Figure 1-2 outlines the structure and flow of the thesis.

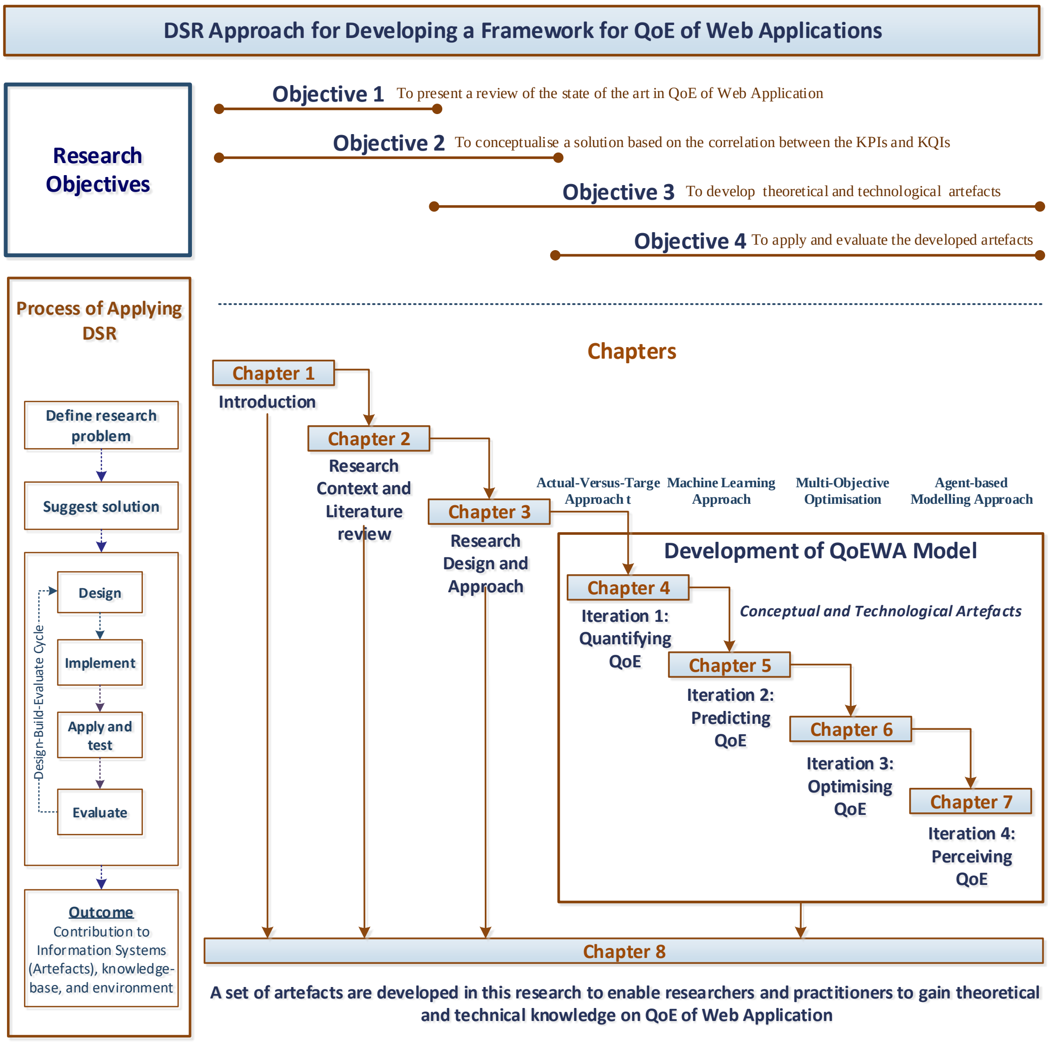


Figure 1-2: Thesis structure

Chapter 2: Literature Review

2.1 Overview

The main aims of this chapter are to: (1) Present a comprehensive introduction to the Web Application and its traditional architecture methodologies, along with the standard software quality models; (2) critically review QoE approaches and expose their limitations from an Information Systems point of view; and (3) to address the challenges of QoE assessment. This chapter is structured as follows: Section 2.2 presents a brief overview of Web Application. Section 2.3 discusses the existing QoE approaches and describes the journey of QoE management, focusing on aspects related to modelling, monitoring, measuring, and optimising of QoE. Section 2.4 discusses the challenges of adopting QoE for Web Application. Finally, Section 2.5 presents an overall summary of the chapter.

2.2 Web Application (WebApp)

In today's dynamic business world, Web Application technology has become an essential element of electronic communications services, especially for commercial enterprises (De Virgilio & Torlone 2009). Web Application or WebApp (Lew et al. 2012) is defined as a "Web system that consists of web server, network, HTTP and browser in which user input (navigation and data input) effects the state of the business" (Conallen 1999 pp. 63). It is a service that allows client and service provider to share and manipulate information electronically over a network service in a wide variety of contexts. The interaction between client and service provider is established when the client (browser) initiates, remotely, a request and the service provider (server) responds accordingly (Finkelstein et al. 2002). In enterprise-scale applications, a high number of lines of code may be required to develop and implement the component of client-provider interaction, depending on the complexity of the business process and technology requirements (Sampath et al. 2007).

In recent literature, it is generally agreed that Web Application is no longer simple and has become increasingly sophisticated, involving complex interactions between several technologies (Lew et al. 2012; Tupamäki & Mikkonen 2013). For instance, Web Application has evolved towards cloud and service-oriented applications that interact dynamically over the internet with intelligent software applications (Chieu et al. 2009). Consequently, with this evolution, the demand of high quality of Web Application has increased considerably and become an important aspect in web engineering. There are many studies have been carried out from different disciplinary perspectives to assess and improve the quality of Web Application. In the literature, however, there are three general approaches for evaluating the quality of Web Application (Mich et al. 2003): (1) Models for evaluating software quality, such as ISO models, e.g. ISO 9126 for quality characteristics and ISO 14598 for process and guidelines; (2) models for usability-focused and human-computer interaction research, e.g. approaches that define quality in term of usability (Seffah et al. 2006; Mifsud 2015); and (3) models for website evaluation and design (e.g. Ramler et al. 2002; Olsina & Rossi 2002; Malak et al. 2004). However, some researchers argue that the quality of Web Application can be also affected by non-technical aspects (Negash et al. 2003; Udo et al. 2010), e.g. governance, customer services and IT support.

Since this research is focused on the adaption of QoE in Web Application, a review of the existing quality models is beyond its scope, but they are described in some detail in Appendix A. Table 2-1 summarises the quality factors extracted from models based on ISO and OASIS (e.g. ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2005; Behkamal et al. 2009; Temnenco et al. 2010; Bevan et al. 2015).

Table 2-1: Summary of the Quality Factors of Web Application

Approach	Factors related to Quality of Service													Factors related to user experience			Aspects of Web Application Architecture				
	Functionality	Reliability	Usability	Maintainability	Efficiency	Portability	Performance	Availability	Accessibility	Success ability	Effectiveness	Satisfaction	Supportability	Expectation	Perception	Satisfaction	HCI	Multi-Tier	MVC	SOA	Enterprise
ISO-9126	√	√	√	√	√	√	X	X	X	X	X	X	X	X	X	X	X	√	√	X	X
ISO -9241	X	X	X	X	√	X	X	X	X	X	√	√	X	X	X	X	√	X	X	X	X
OASIS 2012	X	√	X	X	X	X	√	√	√	√	X	X	X	X	X	X	√	√	√	√	X
TOGAF	√	√	X	√	√	√	√	√	√	√	√	√	X	X	X	X	√	√	√	√	√

In spite of the effort towards improving the quality of Web Application, it is still one of the key debatable issues in web development (Babar 2008; Lew et al. 2012; Kumar & Dadhich 2015). Most of the presented models, which have been derived from standards, such ISO-9126 and ISO-25010, ISO 9241-11, TOGAF, and OASIS or from approaches, such as SOA (Ramler et al. 2002; Calero et al. 2005; Malak et al. 2004; Zeng et al. 2007) do not address the fundamental issues that concern the quality of Web Application from a user perspective. In addition, they fail to provide sufficient insight into the correlation between quality of service and user satisfaction, neglecting that it is pointless to improve quality without considering customers' needs and understanding the complementary relationship between what is offered and what is expected. Consequently, the need to understand the subjective aspects, has led service providers to think beyond objective assessment and to develop a deeper understanding of the subjective assessment, which refers to the user's perception. This challenge addresses the question as to how to improve and optimise quality of Web Applications objectively from a quality perspective so as to enhance user experience (Offutt 2002a; Cecchet et al. 2013; Phillips et al. 2015). However, the interest of this research is adopting a QoE approach, which is generally used for the assessment of multimedia services, from a wider information systems perspective.

2.3 Quality of Experience (QoE)

Quality of Experience QoE is considered as a multidisciplinary concept that interacts with human, information, and technology (Laghari & Connelly 2012; Geerts et al. 2010). It is defined as “The overall acceptability of an application or service, as perceived subjectively by the end-user. It includes the complete end-to-end system effects (client, terminal, network, services infrastructure, etc.)” (ITU-T 2006b pp 1). Given its holistic nature (in the context of the issues highlighted above), QoE is adopted in this study to manage the quality of Web Application and end user experience.

The ultimate goal of the QoE approach is to measure the overall service quality perceived by customers (Baraković & Skorin-Kapov 2013). The measurement of QoE allows a service provider to make an informed decision regarding service delivery and customer satisfaction so to optimise resources (Laghari & Connelly 2012; Menkovski, Liotta, et al. 2009). The measurement of QoE is usually performed by a combination of factors, which are termed ‘objective’ and ‘subjective’ factors (Mitra et al. 2011). Objective factors are typically measured by Quality of Services (QoS) parameters (Brooks & Hestnes 2010), while subjective ones are typically measured by Mean Opinion Score (MOS) tests, which assess how service quality is perceived by customers (Khan et al. 2012), i.e. QoS parameters in Web Application are metrics that can be based on the standard quality models (e.g., (ISO 9241-11 1998; Behkamal et al. 2009), as shown in Appendix A.

The QoE approach traditionally focuses on the domain of multimedia and network services, dealing with multidisciplinary aspects, such as social psychology, engineering science, cognitive science and economics to understand the overall quality requirements (Laghari & Connelly 2012; Baraković & Skorin-Kapov 2013). In terms of QoE management, there are different views on how to model, assess, control, and optimise QoE, emerging from different application domains, as discussed in Subsection 2.3.1.

2.3.1 QoE Management

QoE management focuses on the domain of network services and internet applications, traditionally dealing with three main management aspects: QoE modelling, monitoring and measurements as well as adaptation and optimisation (Baraković & Skorin-Kapov 2013). It is aimed at ensuring that the QoE assessment process is carried out efficiently and effectively (Soldani et al. 2006; Agboma & Liotta 2008). This can be achieved by a continuous-flow process (Marez & Moor 2007), which involves stakeholders with diverse roles and interests (Baraković & Skorin-Kapov 2013), i.e. people from a more operational side have different interests to those with a more strategic focus, as is also the case with those focusing on the design to more management roles (Brooks & Hestnes 2010). This section focuses on QoE aspects from a management point of view, rather than a technical or operational view, i.e. it focuses on the journey of QoE rather than details and destinations related to a specific technology.

Generally, QoE management requires three main processes to achieve the maturity of QoE assessment (Soldani et al. 2006; Hobfeld et al. 2012; Baraković & Skorin-Kapov 2013): (1) Modelling the QoE process to allow the service provider to understand the applications' requirements concerning the service quality and user satisfaction – providing an estimate of the capacity of the network elements and interfaces; (2) monitoring and measuring the QoE process that retrieves technical and operational information regarding network environment, network conditions, terminal capabilities and applications; (3) controlling the QoE process that enables the service provider to react and adjust the quality factors before the user experience is impacted upon. Hence, this process model allows the service provider to improve the overall network service quality as experienced by end-users and to utilise the network resources accordingly. In addition to these processes, a generic framework presented by Agboma & Liotta (2008) and enhanced by Agboma & Liotta (2010) is used to guide the QoE management through seven steps: (1) Characterise the application; (2) design and define the test matrix; (3) specify the test-bed and materials; (4) carry out subjective assessments; (5) analysis of the results; (6) statistical modelling technique; and (7) QoE management strategy. These steps are actually problem-solving ones that give the network operator the ability to apply and demonstrate different QoE management strategies (Agboma & Liotta 2008).

However, due to the multidisciplinary nature of QoE, each application domain has its own QoE requirements and assessments (Dillon et al. 2009; Schumacher et al. 2010), thus some application domains may require sophisticated processes for managing QoE e.g. in emerging domains, where cloud technology is applied and resources are shared over the internet between internal and external stakeholders, it is a challenge to implement QoE management (Hobfeld et al. 2012). In addition, QoE management might be influenced by the nature of the technology and the way it is implemented in the environment (Kulik & Trinh 2011), e.g. under some conditions, technical metrics are not easy to monitor or extract due to technical and/or business considerations (Al-Moayed & Hollunder 2011). Consequently several studies on QoE management have been conducted, particularly where Future Internet technologies and services are involved, e.g. smart information networks and cloud services. This diversity of use makes QoE applicable for this research, where Web Application is considered as a web-based technology runs over the network (Offutt 2002a; Cecchet et al. 2013)

Dillon et al. (2009) presented a QoE management framework based on a user-centric approach that is capable of handling, processing and informing decision-making on the QoE-related parameters and indicators, which are available at all levels, i.e. from the network up to the presentation layers. The framework is implemented to include functions that monitor, measure control, and process data (Dillon et al. 2010) regarding QoS parameters (e.g. performance, availability, reliability, security, etc.) from a network-centric to a user-centric interface to acquire the end-user's perspective regarding an application, rather than relying only upon technical aspects. This enables end-users to be “Always Best Connected” in the networks of the Future Internet (Schumacher et al. 2010), i.e. the user is provided with a service over the internet that can been seen as excellent from the end-user perspective. However, despite the attempts by Dillon et al. (2009) to develop a framework that manages QoE in the context of constructing and managing end-to-end user-centric services, the framework does not construct peer-to-peer infrastructure for exchanging QoE information between users in multi-tiered application architecture, such as a web application deployed in the cloud.

Within this context, Hobfeld et al. (2012) have claimed that the challenge lies in the deployment of the cloud application, where there are multi-providers involved (e.g. players such as database provider, application provider, network provider, etc.), providing different types of services without exchanging the information needed for implementing QoE management, for example: Infrastructure as a Service (IaaS) provides raw network and storage service; Platform as a Service (PaaS) provides an application deployment service; and Software as a Service (SaaS) provides business applications that run on the cloud (e.g. Public, Private, and Hybrid). Each one of these service providers has a perspective on QoE. The core question here is how can QoE be managed across multiple-players? This is actually answered in the context of the peer-to-peer overlay networks approach, where the Internet Service Provider ISP has to provide information to an overlay network (Hobfeld et al. 2012). In the same sense, the overlay approach can be utilised in QoE management to manage the exchange information efficiently between multiple-service providers through a protocol derived from a Service Level Agreement (SLA) that is signed between the players involved in terms of improving QoE for the final service being delivered to the end-user. Furthermore, the overlay approach is not only for exchanging information, for it also can be used for managing the process of QoE optimisation. This is described in the infrastructure presented by De Vleeschauwer et al. (2008), which consists of components in the network core and at the edge to achieve and manage end-to-end QoE Optimisation.

Considering the above and in line with the requirements needed for QoE management, Baraković et al. (2010) presented a multidimensional concept regarding it that provides an overview of QoE aspects, including:

- Technology Performance aspects that are based on factors, such as availability, reliability, quality, effectiveness, latency, etc. on four levels from infrastructure to presentation (i.e. application/service, server, network and device);
- Usability aspects, which are related to efficiency and effectiveness regarding human interaction with applications running on a terminal e.g. the browser
- Subjective Evaluation aspects, such as performing subjective evaluation of the application/service (e.g. response time, content personalisation, complexity, and attractiveness), network (e.g. ease of access, speed, security and interoperability) and device (e.g. CPU, memory usage, display related issues, and the interface);
- Expectation aspects that are related to subjective evaluation, i.e. understanding users' expectations in relation to the quality of the delivered service;
- Context aspects, such as variables related to the environment, personal/social context, and technological context as well as culture, e.g. demographic information.

It can be hence concluded that Baraković et al. (2010) implicitly presented a classification framework that provides a wide range of technology-centric and user-centric aspects as well as metrics for understanding the requirements and expectations needed for the management of QoE.

In contrast to the management frameworks discussed above, Kim et al. (2010) presented a QoE management framework, called the "in-service feedback QoE framework" that involves end users in QoE management studying the reason for service dissatisfaction, rather than satisfaction, as in the model proposed by Dillon et al. (2009), which ensures that user is 'Always Best Connected' in networks. Four aspects for QoE management are considered (Kim et al. 2010), including: (1) Improving service quality in term of usability; (2) finding the reason for quality detraction by involving end users as participants in the process of measuring the service quality; (3) monitoring the network states by an in-service feedback schema that buffers the feedback information from routers, applications, and users for a fixed short time period, until the analysis is performed; and (4) evolving the user's expectation and experience through a sustainable model based on the feedback, which is developed continuously. Involving end users in the process of examining the detraction of the service, allows the service provider to reduce the cost of QoE assessment. In terms of resource management, the process fails to address the steps needed for QoE optimisation. However, this limitation has been considered by Staehle et al. (2011), who presented a model based on the Aquarema approach (Application and Quality of Experience-Aware Resource Management) as a solution for managing and adopting network resources dynamically with respect to user satisfaction.

Building upon the literature outlined above, it can be concluded that QoE management is not a task that is achieved at a point in time or a function key that is used to perform its assessment of. Rather, QoE management is a journey that involves collaborative processes between stakeholders in a network environment. Since each environment has its own characteristics, technology, processes and stakeholders, it is important to understand how these elements interact to each other as well as the context of QoE in terms of where, when and how it is applied. This is discussed in more detail in the next sections.

In summary, Table 2-2 compares the QoE management frameworks discussed above in terms of modelling, measurement, monitoring control and optimisation as well as implementation. It can be seen that QoE modelling is an essential element for constructing a QoE management framework, while the implementation is not often considered in most existing frameworks. Furthermore, most of the these frameworks have been applied in relation to multimedia and network services, thus neglecting other domains, such as web applications based on multi-tier and SOA architectures. Since this research focuses on the information systems environment, DSR is utilised as the methodology (discussed in Chapter 3) for developing the QoE framework that includes theoretical and technical artefacts i.e. the theoretical artefacts reify the aspects related to modelling and methods of measurements, monitoring, and optimisation, while the technical artefacts focus on the implementation aspects.

Table 2-2: Summary of QoE Management Frameworks

QoE Management Frameworks	QoE Management Aspects						Approach Applied	Application Domain
	Modelling	Measurement	Monitoring	Optimisation	Assessment	implementation		
(Hobfeld et al. 2012)	√	√	√	√	√	√	Overlay network management	Cloud and Multimedia Applications
(Baraković & Skorin-Kapov 2013)	√	√	√	√	√	x	Multidisciplinary Approach	Generic QoE Framework for Wireless Networks Services
(Laghari & Connelly 2012)	√	√	√	x	√	x	Ecosystem Approach	Ecosystem & Multimedia Applications
(Dillon et al. 2009)	√	√	√	x	√	x	User-centric and Always Best Connected (ABC) Approach	Internet Connection
(Baraković et al. 2010)	√	√	√	x	x	x	User-centric and technology-centric approaches	Next-Generation Network (NGN)
(Kim et al. 2010)	√	√	√	x	√	x	In-service feedback approach and multi-agent model	Distributed Applications
(Staehle et al. 2011)	√	√	√	√	√	√	Application and QoE aware resource management (Aquarema) approach	YouTube in Wireless Mesh Networks
(Soldani et al. 2006)	√	√	√	√	√	x	QoS and QoE management, ITU, and other multidisciplinary approaches	Universal Mobile Telecommunication System

2.3.2 QoE Modelling Process

Generally, for developing efficient and robust modelling, the following key principles are considered: (1) Producing a static structure or architecture of the model using objects and relationships; and (2) constructing the dynamic behaviour of the model by showing collaboration among the objects (Schalles 2013). In most QoE studies, modelling tasks are achieved by developing architectures that describe objective and subjective factors, metrics, data and their relationships, mappings and interactions. Based on these perspectives, QoE modelling is discussed in this section.

With respect to the multidisciplinary context of QoE, Kilkk (2008) presented a model for QoE based on a communications ecosystem approach, which is defined as "an approach that incorporates different disciplines such as technology, business, context, and human behaviour" (Laghari & Connelly 2012). Hence, Kilkk's model describes the interaction between three main domains (business, technical, human), each of which has its own terminology and languages, i.e. stakeholders from each domain use their own terminology to discuss their concerns with each other, but a terminology from one domain is not directly applied to other domains. For example, in a business domain, people are concerned about revenue; in a technical domain, people are concerned about performance; and in a human domain, people are concerned about experience. These domains are expressed as an ecosystem service that interfaces the relationships between those multiple domains. However, they are not associated with a contextual domain that encompasses aspects of external factors, such as the environment or social context. In addition, there is a lack of taxonomies on the factors that influence the interfaces between human and technical domains as well as between human and business domains. This may be overcame by the taxonomy of QoS and QoE, which is presented by Moller et al. (2009) for multimodal human-machine interaction. It includes three layers: (1) QoS-influencing factors, which are connected to the user, system and the context of use, e.g. environmental and service factor; (2) QoS interaction performance aspects, which are related to user behaviour and system performance; and (3) QoE aspects, which include factors related to the quality perceptions, e.g. acceptability and usefulness. The relationships between layers can be one-to-one or one-to-many, depending on the domain of the system, user and context. Each layer is provided with metrics that are comparable across the stakeholders who are involved in the assessment. However, despite that these metrics making the assessment more systematic, they are not exhaustive due to the lack of consideration of the business aspects (Laghari & Connelly 2012).

Consequently, Laghari & Connelly (2012) presented a QoE model in a communication ecosystem, encompassing multidisciplinary aspects, including: Technology, business, human behaviour and context. The model is based on a high level structure and low level details to be adapted to various contexts. It consists of four domains (human, technology, business, and context): The human domain is an entity that determines the demographic factors, which influence the interaction between the user and technology. The technological domain is a blueprint that represents all technological aspects, including design, implementation, deployment and service delivery. It is connected to QoS parameters to determine the quality of services and network resources. The business domain is a holistic view of business aspects, describing the interaction between the user and provider in terms of experience, pricing, advertising, promotion and churn rate. The contextual domain is developed as a communications ecosystem that represents the situations and environment during the interaction between humans, business, and technology. These domains are mapped to establish the relationships between dependent and independent factors. The model was applied in a VoD service to provide a taxonomy of QoE parameters and their interactions to help practitioners in thinking creatively and more broadly about QoE. However, the application does not include a QoE-based test that conducts a user study for validating the model.

With all of the above in mind, Song et al. (2012) presented a framework that organises the influencing factors of QoE into three main components: (1) User component, which is based on factors that are related to perception, motivation, user profile, needs, expectations and emotions; (2) system component, which relates to the overall performance of the service from two perspectives, sender and receiver; and (3) context component, which includes four elements (users, device, network and mobile video service), interacting physically over the network infrastructure. The framework encompasses many factors for measuring and optimising the user experience, but it does not capture the correlation between these factors. In addition, some influencing factors, such those relating to user experience, may need more detailed and empirical research to characterise their impact on QoE optimisation.

In order to capture the correlation between factors, their relationships need to be quantified. For instance, the relationship between user component and system component was considered by Perkis et al. (2006), who developed a model for quantifying the metrics of QoE evaluation. It is conceptualised in a tree-based topology with two main branches: Measurable and non-measurable parameters. The former relate to the technology of the service (e.g. terminal software and hardware, QoS parameters, codec evaluation and content evaluation.). The latter pertain to the user experience (e.g. expectations, understanding and satisfaction, and attitude and habit). The model is used as a tool for modelling the user experience of mobile multimedia services, i.e. it measures attitude, reliability and audio video quality using Mean Opinion Scores (MOS). However, the assumption of assigning every non-measurable parameter to users is not likely to be efficient in the case where they are actually related to technical aspects. Accordingly, some measurable parameters are not necessarily associated with technical aspects. The relationship between factors has also been distinguished by other researchers as technical and non-technical aspects (Soldani et al. 2006), i.e. technical factors are mainly related to objective aspects, while non-technical factors pertain to subjective aspects.

The correlation between factors related to human experience and technical parameters was considered by De Moor et al. (2010), who proposed a model for evaluating QoE in a mobile, testbed-oriented, living lab setting. The model has three main logical entities, reflecting the interdisciplinary approach: QoS monitoring entity, the contextual monitoring entity and the user experience monitoring entity.

The implementation of the model is based on a component based architecture that consists of a distributed system, including the following main components: Controller engine, monitoring probe, logger, user module, management and platform abstraction Layer. These components work together as a tool in a real-time environment for monitoring QoS, context information, and subjective experience. The model was refined by Geerts et al. (2010), whereby additional measurement tools were added, consisting of four main components: User, Information and Communications Technology (ICT) product, use process and context. The components are integrated for developing a detailed and comprehensive model that provides more detailed insight into user aspects. The development process is carried out by several interdisciplinary workshops that bring together researchers and practitioners from different backgrounds (e.g. sociology, communication science, cognitive, anthropology, software engineering, human-computer interaction and product design) to enable participants to integrate user oriented and technical definitions.

On the other hand, with the recent shift towards Over-The-Top service (OTT) services, which is defined as a platform that operates and distributes multimedia service and other web services over the internet, Rivera et al. (2015) have focused on modelling QoE in web-based OTT services. Others have applied OTT services in real-time multimedia services (Seppanen & Varela 2013; Bouten et al. 2013). Rivera et al. (2015) presented a framework that includes functional and non-functional requirements. The former are described as input and output parameters that are associated with states and transitions, consisting of a set of conditions. If the conditions are satisfied, the transition is initiated, turning the state from the current state to the next state (This concept is known as the Extended Finite-State Machine EFSM). The updating functions of these states capture functional possibilities of the interaction between the users and service. They also specify the values of the non-functional parameters, which are expressed as objective, subjective, and business parameters. The framework provides details on the interaction between these parameters, as well as specifying how they can influence the user's perception of the service dependability. Within the same context of OTT services, Seppanen & Varela (2013) proposed a network access point model that incorporates three aspects: Customer, user and application. The customers are classified into two types of users: Normal users (who have subscribed to a best-effort Internet connection service), and premium users (who have subscribed to a high-end connection service). This is to be sure that premium users always receive the highest quality of service, whilst respecting that normal users' experience does not deteriorate. It was found by the authors that the access point solution is efficient for optimising resources, according to the type of users. However, the classification sometimes fails in identifying applications that run on the same webpage and share similar statistical features (e.g. HTTP download and HTTP video streaming).

For modelling QoE, ITU-T (2014) presented a recommended framework for the QoE of web-browsing. The framework provides guidance in the development of an opinion model, overview of key influence factors and a starting point for performance assessment. The framework includes a taxonomy of influence factors (IF) and related model parameters. The former are grouped into three main categories: User influence factors, context influence factors, and system influence factors. The user influence factors are not addressed in sufficient detail in ITU-T (2014). However, the context influence factors are defined as factors related to location, interactivity, task type and task urgency (without time constraints). While the system influence factors are divided into sub factors, including: Server-related, content-related, delivery network and client influence factors. The interaction between these factors is represented as delivery chain process for a typical webpage, which presents the scenario where a user requests a webpage or clicking on a link. As a result, the downloading of the content will be initiated and then fetched and rendered by the functions of the browser. The framework also provides details on the perceptual events in a webpage view cycle from the end user point of view, including the: Time when the user requests a webpage; time when a change in the status happens; time when the first element of the requested webpage appears; time when page is sufficiently rendered; time when the initial webpage request is sent;

time when the webpage is processed; time when all objects of the webpage are downloaded from the server; and the time when the webpage is completely rendered and displayed (Yamauchi & Ito 2015).

Based on the models discussed above and with reference to the Web Application architecture, there an opportunity in this research to integrate QoE architecture with Web Application architecture to extract QoE factors, i.e. QoE components of user and service could be mapped with the component-based application architecture, which forms the interaction between the consumer and provider (Lima & Carvalho 2011; Gonzalez et al. 2013), i.e. factors related to users (e.g. overall assessment of users' needs, feelings, performance and intentions.) can be derived from the presentation layer. Factors related to business functions can be derived from the business logic layer, and so on. Furthermore, Reichl et al. (2013) proposed a framework that aims to improve QoE modelling and prediction for mobile broadband services (e.g. mobile Web browsing and file download). The framework is based on a layered approach distinguishing between the user, application and network layers. Each layer has its own performance indicators, which are derived from user studies and technical analysis. The modelling of these layers is considered as an application/service specific solution, where the characteristics of each application/service have their own technical performance and user experience.

In sum, it is clear that there are various perspectives in regards to the meaning of the term of QoE modelling, with some researchers having presented well-structured detailed taxonomies of QoE (e.g., Moller et al. 2009; Laghari & Connelly 2012), whilst others have addressed the multidisciplinary nature of QoE by, focusing on three domains: Business, technical, and human (Kalevi Kilkk 2008; Laghari & Connelly 2012). Perkis et al. (2006) developed a model based on a tree-based topology for quantifying the metrics. Song et al. (2012) presented a components-based framework and De Moor et al. (2010) proposed a model based on logical entities, thus reflecting the interdisciplinary approach. Finally, Reichl et al. (2013) considered a layer-based approach. It can be concluded that these models have been developed as solutions for particular problems and technologies. Consequently, QoE modelling may vary from business to business and from one technology to another. Table 2-3 summarises QoE modelling approaches discussed above, taking into account the goal question metric approach proposed by Basili et al. (1994) and applied later by Lilburne & Khan (2004) in QoE for identifying the factors, metrics, data and relationships.

Table 2-3: Summary of QoE Modelling

Modelling Approach	QoE Modelling Aspects						Approach and Method Applied	Application Domain
	Data Source	Metrics	Factors	Obj. Metrics	Subj. Metrics	Relationships		
(Kalevi Kilkk 2008)	x	x	√	√	√	x	Ecosystem Approach	Communications Ecosystem
(Moller et al. 2009)	x	√	√	√	√	x	Multimodal human-machine interactions	Generic QoE Model
(Song et al. 2012)	√	√	√	√	√	x	Organises the influencing factors into three components: User, System, and Context	Human-Computer-Interaction in Mobile Video
(Perkis et al. 2006)	x	x	√	√	√	x	Measurable and non-measurable parameters for quantifying QoE	Multimedia Services
(Soldani et al. 2006)	√	√	√	√	√	√	QoS and QoE management, ITU and other multidisciplinary approaches	Universal Mobile Telecommunication System
(Geerts et al. 2010)	x	√	√	√	√	x	Multidisciplinary, including technical and user aspects.	HCI in Information Communications Technology
(Rivera et al. 2015)	√	√	√	√	√	x	Functional and non-functional evaluation	OTT and services across IP networks
(Seppanen & Varela 2013)	√	√	√	√	√	x	Premium users always receive highest QoS	OTT multimedia services
(Yamauchi et al. 2015)	√	√	√	√	√	√	Regression analysis that relates TCP to the user-satisfaction	Online Shopping Web Services

There have been several attempts in the literature at modelling QoE, however, most examine QoE objective and subjective factors at an abstract level, i.e. they often focus on modelling objective and subjective factors without addressing their relationships and dependences. It is also clear that most of the proposed QoE models have been in the context of multimedia and network services, whilst the QoE modelling aspects are more related to web engineering and information (Hsu et al. 2013; Upadhyaya et al. 2014). This provides the motivation for this research to employ QoE with consideration of the wider Information Systems aspects (discussed in Chapter 3).

2.3.3 QoE Monitoring Process

QoE monitoring is a vital part of the QoE assessment process that ensures the compliance of service quality with user needs by correlating and evaluating various objective factors along with subjective factors (Lima & Carvalho 2011). These factors are broadly decomposed into criteria and metrics, as proposed by Seffah et al. (2001), to describe the service quality as viewed from the end-user's perspective (Skorin-kapov 2012). Hence, the QoE monitoring aims to analyse the values of the obtained metrics and thereafter, assessing QoE. This can be achieved by addressing the following four questions (Baraković & Skorin-Kapov 2013): (1) What are the metrics to be collected?; (2) Where are the metrics' points to be collected from?; (3) When are the values of the metrics to be collected?; (4) How are the metrics going to be analysed and measured? (Hoßfeld et al. 2007). The following subsections discuss these questions in more detail.

- **QoE Monitoring Metrics**

Building upon the approaches discussed earlier in the modelling section (e.g. Moller et al. 2009; De Moor et al. 2010; Laghari & Connelly 2012; Song et al. 2012; Skorin-kapov 2012), QoE metrics are generally expressed by a set of technology-platform dependent parameters, i.e. metrics are traditionally based on the type of the service and its context and technology. However, they can be theoretically categorised into two types, objective and subjective, that correspond to the human, technological, contextual and business domains. The correlation between these categories represents the measurement of QoE (Alreshoodi & Woods 2013).

Objective metrics are parametrically driven by a set of quantitative and technical measures that represent application and network factors related to system resource characteristics (Habachi et al. 2012). Application factors are determined by metrics such as codec, bit rate, echo cancellation, loss concealment and so on, while network factors are determined by metrics such as throughput, delay, jitter buffer, bandwidth, loss rate, packet loss, latency and so on (Skorin-kapov 2012). However, ITU-T (2005) define the objective metrics as QoS parameters that describe latency (delay), bit error rate, data loss rate jitter, bandwidth and so on. In terms of Web Application and web-browsing, ITU-T (2014) defines the following objective metrics:

- ***t0:*** The response time of requesting a new web page, e.g. time between pressing enter on the URL bar and loading the web page;
- ***tSBr:*** The time taken when a change in the status/progress bar occurs;
- ***tSgB:*** The time taken between the viewed web page disappearing and loading the requested web page;
- ***tPrs:*** The time taken between when the first object of the requested page is loaded and rendered on the screen;
- ***tPPLT:*** The time taken when the requested information is rendered and the user accesses it;
- ***tVSrc:*** The time taken when the web page (as determined by browser windows size) is fully displayed;
- ***tIHRs:*** The time taken when the original HTTP request is processed and sent by the browser;

- ***tBHPr:*** The time taken when the head element of the HTML page is received;
- ***tHp:*** The time taken when the HTML page is being processed for display;
- ***tTPLT_1:*** The time taken for loading all the requested objects of the web page from the server to the client side;
- ***tTPLT_2:*** The time taken for a web page to completely display in the web browser.

Subjective metrics are often qualitative in nature and used to measure the level of service quality perceived by the end user, considering factors related to effectiveness, efficiency, enjoyment, social presence, satisfaction, impression, plausibility and so on (Brooks & Hestnes 2010). When defining the subjective metrics, it is important to understand the end-user's expectation and perception of the service quality along with the objective measurements (Soldani et al. 2006). For example, in order to evaluate the efficiency of the performance from the user's point of view, a subjective question needs to be formulated to rate the time taken to complete a particular task. This question can be taken as a metric and the answer indicates its value. In this context, an attempt was made by Schatz et al. (2011) to bridge the gap between the acceptancy and the objective measurements, whereby three essential questions were formulated as follows in order to define the subjective metrics:

1. How should acceptability metrics be defined? ;
2. How to evaluate and measure acceptability?; and
3. How do the ratings of acceptability relate to objective measurements?

These questions can be used to construct a survey or a test (e.g. with a Likert-scale) that evaluates the user's experience using an appropriate QoE scale.

- **QoE Monitoring Points/Nodes**

QoE metrics data can be collected within the network and/or at the edge nodes of the network, e.g. client/server terminals,, or application (Baraković & Skorin-Kapov 2013). For instance, the edge nodes may generate data from a distributed architecture, as in De Moor et al. (2010), where (1) QoS monitoring entity monitors the technical objectives parameters from four monitoring points: Device, infrastructure, network and application; (2) a contextual monitoring entity deals with the context of the application, i.e. location, mobility, sensors and operating system; and (3) an experience monitoring entity that deals with the subjective data relating to customer satisfaction. The monitoring points collect two sets of data: Objective and subjective (Brooks & Hestnes 2010).

Monitoring objective data is a task that fluctuates from business to business and from technology to technology, i.e. monitoring can be performed in different ways depending on the scope and purpose (Al-Moayed & Hollunder 2010). Keeping the focus of this research in mind, however, QoE monitoring points are investigated from the aspects of Web Application engineering and in line with multi-tier architecture. For instance, in the context of Web Application architecture, performance metrics can be monitored from a back-end database server, middleware server, or from a client network. Hence, since monitoring points are chosen in accordance with sophisticated and heterogeneous metrics, different tools and techniques are used to collect and analyse the monitored data. Many technological resources, e.g. operating systems and middleware software, are provided with compatible tools for monitoring QoS metrics, but few consider quality and standards (Al-Moayed & Hollunder 2010). Hence, service providers are increasingly required to develop bespoke tools installed and run behind the server or client to monitor additional custom metrics, e.g. Web Service Level Averment (WSLA) has been introduced by IBM to monitor QoS metrics in Web Services (Benveniste et al. 2008).

On the other hand, monitoring subjective data is an extremely challenging task, because it fluctuates from individual to individual across social groups and societies. Data related to the user's perception, experience, and expectation can be captured directly from end users (Baraković & Skorin-Kapov 2013) in many ways, including surveys and questionnaires (Menkovski, Liotta, et al. 2009; Menkovski, Exarchakos, Liotta & Sánchez 2010; Skorin-kapov & Barakovic 2015), in a laboratory environment (Gardlo et al. 2011; Yamazaki et al. 2012; Guse et al. 2014), through quick polling and by an interactive polling session (Soldani et al. 2006). In addition, data can be extracted from the server side, e.g. Customer Relations Management (CRM) databases (Mushtaq et al. 2012) or social networks and data mining across disciplines (using data mining or machine learning techniques) (Menkovski, Exarchakos & Liotta 2010; Witten et al. 2011). Some general recommendations regarding metrics interfaces have also been presented by the QoE community (ITU-T 2005).

- **QoE Monitoring Intervals/Schedule**

An important question in monitoring, after defining the metrics and their monitoring edges, is when to monitor QoE? It is wise to monitor it on a regular-interval basis to ensure that service is delivered with the highest quality and meets user requirements. QoE can be monitored before, during and after the touchpoints (Baraković & Skorin-Kapov 2013; COMBO 2013).

- Before the touchpoint, service and resources need to be checked and optimised prior to offering it to the customer.
- During the touchpoint, service needs to be monitored and controlled for SLA compliance.
- After the touchpoint, service needs to be evaluated and improved, if needed, to reduce churn.

The time-interval for monitoring the service is varied based on the nature of the deployed technology, business requirements and the environment. Monitoring can be performed in: (1) Real-time, where metric values are analysed and computed instantaneously, as performed by Mushtaq et al. (2012) and Seppanen and Varela (2013); (2) near-time, where metrics values are monitored at set intervals (e.g. time triggering or batch) rather than instantaneously; and (3) some-time, where metrics values are computed only once, e.g. ad-hoc and manual monitoring performed in a laboratory environment. Due to the advanced technologies, objective metrics can be monitored and computed in real-time giving fast and quantitative results, in contrast to the monitoring of subjective metrics, which often involve time-consuming and expensive tasks. Several QoE predictive models have been proposed for facilitating the monitoring process of the subjective metrics (Menkovski, Exarchakos, Liotta & Sanchez 2010; Khan et al. 2012; Balachandran et al. 2013).

Table 2-4 summarises the QoE monitoring approaches discussed above, according to the four questions designed by Baraković & Skorin-Kapov (2013) and mentioned earlier in this section. They addressed the following monitoring aspects:

- QoE metrics to be monitored.
- QoE monitoring nodes and location.
- Intervals and timing of QoE monitoring.
- Analysis and interpretation of the monitored metrics.

Whilst most of the discussed monitoring models have had success in defining the metrics and their monitoring edges, they have failed to control when to initiate and invoke the monitoring process. However, this limitation is overcome in other studies related to QoE optimisation (as presented in Subsection 2.3.4). Furthermore, it has been found that the monitoring process varies depending on the technology of the operated service.

Table 2-4: Summary of QoE Monitoring Models

Monitoring Approach	QoE Monitoring Aspects				Target		Approach and Method Applied	Application Domain
	Metrics	End-point Nodes	Intervals	Analysis	Obj. Measures	Subj. Measures		
(Lima & Carvalho 2011)	√	√	√	√	√	√	Self-adaptive service management / SOA	Multimedia and Network Services
(Hoßfeld et al. 2007)	√	√	√	√	√	√	IQX hypothesis - Defining relationship between QoE/QoS	Edge-Based Applications /VoIP
(Alreshoodi & Woods 2013)	√	√	√	√	√	√	IQX hypothesis – layering QoS and QoE	Multimedia Service
(Skorin-kapov 2012)	√	√	x	√	√	√	Multidimensional generic model - ARCU	Generic Model
(Brooks & Hestnes 2010)	√	x	x	√	√	√	Statistical techniques	QoE Data in the Network and Media Services Industry
(Schatz et al. 2011)	√	x	x	√	√	√	SOS hypothesis - standard deviation of opinion Scores	Web Pages /Web Traffic
(Al-Moayed & Hollunder 2010)	√	√	√	x	√	x	SOA, UDDI and WS-Policy	Web Services
(Benveniste et al. 2008)	√	√	√	√	√	x	QoE contract monitoring technique	Web Services
(Mushtaq et al. 2012)	√	√	x	√	√	√	Prediction using ML / Extracting data from CRM	Web and Database Service
(Guyard & Beker 2010)	√	√	√	√	√	√	A real-time basis by the SLA management tools / Anomaly detection	Voice over Internet Protocol (VoIP) Service

2.3.4 QoE Measuring Process

In QoE, the measuring process computes and evaluates the obtained objective data from the user’s perspective, in contrast to the ISO/IEC TR 9126-3 (2002), OASIS (2012) and Susila & Vadivel (2014) approaches discussed earlier, which normally assess service quality from a relatively technical and objective perspective. In fact, these approaches are inherently interdependent, but the distinction is that the QoE approach fundamentally focuses on the telecommunication technology, while the quality models based on ISO 9241-11 (1998) entirely focus on web technology. In addition, QoE is measured by objective and subjective methods

Objective measuring methods compute and evaluate the objective data, which is obtained by the monitoring process (Schatz et al. 2013). Many methods have been proposed for different applications and environments. Statistical analysis and machine learning models have applied to measure the quality of services deployed and installed on mobile and TV platforms (Menkovski, Exarchakos, Liotta and Sánchez 2010). Geerts et al. (2010) developed a compressive framework for measuring and predicting services and products of Information and Communications Technology (ICT). Balachandran et al. (2013) developed a model for internet video services that measures metrics such as average bitrate, join time and rate of buffering. Gardlo et al. (2011) used statistical methods and social networks to evaluate and measure QoE for HD Video Streaming. Obafemi et al. (2011) conducted an analytic and experimental study to evaluate how the E-model in Voice Over-IP (VoIP) quality is impacted upon by the jitter playout buffer. Many other researchers have developed QoE methods to measure, predict and evaluate multimedia services and network services. Most of these models are applied to multimedia and telecommunication services and focus on the QoE metrics which are standardised by ITU-T.

Subjective measuring methods are based on the monitored subjective data and the nature of the service to be measured. ITU-T (2006) have provided guidelines and standards on how to choose and set the methods, i.e. method environment, test requirements, service type, rating, scoring and scale (Schatz et al. 2013). The most commonly used method is MOS, which is used for measuring the quality of telecommunication and network services. MOS is a standard in QoE, with suffix LQ referring to Listening Quality, TQ to Talking Quality, CQ to Conversational Quality, S to Subjective factor, and O referring to Objective factor (if it is measured from the user's point of view) (ITU-T 2006b). MOS is based on an ordinal scale of five-points: (1) Bad; (2) poor; (3) fair; (4) good; and (5) excellent. Besides MOS, other methods and scales are also available for other purposes, e.g. Bark Spectral Distortion (BSD) is used for psychoacoustic and visual experiments. The Perceptual Speech Quality Measure (PSQM) is used for measuring voice quality and Video Quality Measurement (VQM) is used to assess video streaming quality. Measuring Normalising Blocks (MNB) are used for evaluating speech quality.

These methods, including MOS, are actually measured on nominal and ordinal scales, which represent categories and differentiate between characteristics, without relying on the interval or ratio level. Consequently, a label-free scale (Brooks & Hestnes 2010) or "labels only at the end of the scale" is recognised and used with a ratio of a ten-point scale starting from 0-extremely low to 10-extremely high. Ideally, the different scaling methods will allow for various statistical techniques to be applied, especially for mapping the objective and subjective measurements (Brooks & Hestnes 2010). For example, a nominal scale allows subjective attributes to be categorised. Whilst a ratio scale allows various types of objective and subjective attributes to be correlated and compared using the same scale, but of course with different weights. That is, each attribute should be assigned with a percent weight to reflect its impact and importance within the overall measured QoE value (Olsina & Rossi 2002; Chang et al. 2010).

Mapping between the objective and subjective measuring methods is an essential and important task in the life-cycle of QoE assessment (Hobfeld et al. 2012). Otherwise, what is the point of measurements, if the relationship between service quality and user satisfactions is not expressed? There are several QoE mapping approaches that have been designed for multimedia and network services (Alreshoodi & Woods 2013) and most of them involve applying statistical techniques for mapping QoE parameters. For example, the QoE-aware QoS management model proposed by Agboma & Liotta (2008) statistically correlates QoS parameters with QoE perceptions to estimate the influence degree of the measured QoS factors on user perceptions. The finding implies that statistical techniques, such as correlation and regression analysis, are efficient when the realistic objective and subjective data are available. However, they fail when there is missing or incomplete data due to the technical difficulty of collecting all or some of the metric values needed for measuring QoE. In addition, collecting up-to-date realistic data for statistical analysis is an expensive and tedious process. Ideally, for maximum currency, the QoE measuring process should continuously poll customers on their experience. Consequently, researchers have started to look toward models that intelligently assess and predict QoE metrics values. Several types of predictive models have been developed for predicting QoE with respect to the application domain, technology, metrics and data availability (Baraković & Skorin-Kapov 2013; Aroussi & Mellouk 2014).

According to a survey conducted by Aroussi & Mellouk (2014), most of the QoE prediction models are based on the machine learning (ML) approach and use an inductive supervised learning approach, where the predictive rules are generated from particular observations or learning. The QoE-QoS relationship is categorised into two models: (1) Offline batch model based on offline learning, in which training data is available during the batch training process for determining the relationship between the objective (inputs) and subjective parameters (outputs). Generally, the offline batch models use regression analysis and classification methods (2) an online increment model is used to estimate QoE from QoS metrics in real-time, without conducting subjective studies, e.g. a survey. In spite of the fact that most of the predictive models are developed in an offline batch mode (Aroussi & Mellouk 2014), there have also been attempts to adopt the online increment model for assessing QoE as in Menkovski, Oredope et al. (2009) and Menkovski, Exarchakos & Liotta (2010). However, predictive models based on offline batch mode or online increment mode cannot be generalised for all internet and network services, because each service is associated with a set of quality parameters reflecting a particular context.

Most of the current mapping approaches can be categorised into two approaches (Alreshoodi & Woods 2013): (1) A top-down approach based on data collected from the user-side; and (2) a bottom-up approach based on data collected from the server-side and related to QoS. Both approaches can be applied alongside each other in complementary ways, depending on the data availability and the degree of association between the QoE parameters. Figure 2-1 illustrates the two approaches.

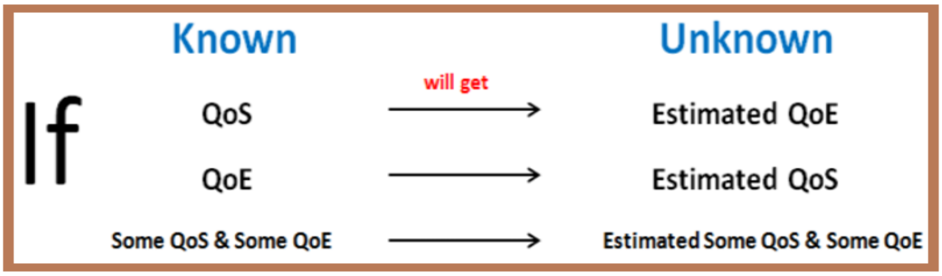


Figure 2-1: The mapping approaches Source:(Alreshoodi & Woods 2013)

However, the interactions between the objective and subjective parameters are still insufficiently understood and there are no standards or guidelines that systematically quantify and map these parameters. Most of the mapping approaches based on the predictive models are only partially developed and fail to consider the overall QoE life-cycle in a holistic manner (Alreshoodi & Woods 2013). Hence, It is still a challenge to quantify and map the relationship between objective and subjective parameters (Alreshoodi & Woods 2013; Aroussi & Mellouk 2014). This limitation is considered in this research and a suggested solution is presented in Chapter 4 and Chapter 5.

Table 2-5 summarises the QoE measuring approaches discussed above, considering the objective and subjective measurements, mapping between measurement, weighting and scale of measurements as well as the relationship between the objective (inputs) and the subjective parameters (outputs), i.e. whether it is based on either an offline batch or an online increment. Whilst there have been various attempts to develop objective and subjective measurement methods, very few researchers have figured out how to quantify the measurements, as well as bridging the gap between the two types of measurement.

Table 2-5: Summary of QoE Measurement Models

Measurement Approach	QoE Measurement Aspects						Approach and Method Applied	Application Domain
	Obj. Measurement	Subj. Measurement	Mapping	Weighting & Scaling	Offline batch	Online Increment		
(Menkovski, Exarchakos, Liotta & Sánchez 2010)	√	√	√	√	X	√	Online QoE prediction – machine learning	Mobile Multimedia Streaming Content
(Hsu et al. 2013)	√	√	√	√	√	√	Capture user's perception by MK-means and SVM algorithm	Web-based Applications
(Balachandran et al. 2013)	√	√	√	√	X	√	Machine learning / Decision Tree	Internet Video Applications
(Gardlo et al. 2011)	√	√	X	X	√	X	Social networking and subjective testing of	HD Video Streaming
(Obafemi et al. 2011)	√	√	√	X	√	X	ITU-T recommended quality measurement model	E-model in VoIP Applications
(ITU-T 2006b; ITU-T 2014)	√	√	X	√	√	X	ITU-T recommended quality measurement model - MOS	Web Browsing
(Hobfeld et al. 2012)	√	√	√	X	X	√	Overlay adaptation – (Interaction between players)	Cloud Applications
(Alreshoodi & Woods 2013)	√	√	√	√	√	√	Survey on QoE QoS/QoE correlation models	Multimedia Services
(Agboma & Liotta 2008)	√	√	√	√	√	X	A statistical modelling technique correlates QoS/QoE	Streaming of Multimedia Content
(Aroussi & Mellouk 2014)	√	√	√	√	√	√	Machine learning-based on QoE-QoS correlation models	Web Surfing, Video Streaming, VoIP, and VoD

2.3.5 QoE Optimisation and Controlling Process

The ultimate aim of QoE management, following modelling, monitoring and measurement is to optimise and control it (Baraković & Skorin-Kapov 2013). The QoE optimisation process plays a central role in utilising network resources efficiently and delivering the desired user satisfaction effectively (Soldani et al. 2006). From a service provider perspective, the goal is to maximise the perceived user experience and minimise the technical resources dynamically at an acceptable level of satisfaction in accordance with user requirements or SLA. From an end-user perspective, the expectation is to receive a service with high quality and at reasonable cost (Batteram et al. 2014). The optimisation process aims to promote a balance between the two perspectives. Referring to the aspect of managing the control and optimisation of QoE, Wamser et al. (2013) presented a cross-layer architecture solution to manage resources for video streaming. The architecture includes the following main entities: (1) Video application monitor; (2) network and flow monitor; (3) network advisor; and (4) resource management actions. These entities work together to reduce the operational expenditure, on the one hand and to increase the quality on the other (especially the user-perceived quality of the applications used in the network). Linking this architecture with the resource management strategy, Wamser et al. (2013) presented two of several possible control actions, i.e. network and service. The former is taken when the goal is to optimise the overall QoE, with all possible resources to be utilised without considering the cost. The service control action is taken when the goal is to reduce the cost, without considering quality of service perceived by the end-user. This insight presented here, leads to an answer to the question raised by Drogseth (2009) “How to Pick the Right QoE Solution for You?”, as it provides a solution to the problem of where to concentrate the diagnostic efforts towards QoE. In order to implement a successful optimisation and controlling strategy, four key elements need to be identified (Baraković & Skorin-Kapov 2013): (1) QoE optimisation indicators, answering what to optimise?; (2) QoE optimisation points, answering where to optimise?; (3) QoE optimisation initiatives, answering when to optimise?; and (4) QoE optimisation methods, answering how to optimise?.

- **QoE Optimisation Indicators (KPIs and KQIs)**

Identifying QoE optimisation indicators is a task based on the QoE measurement process, where the actual values of the QoE factors are computed and evaluated and then fed forward to the optimisation process (Baraković & Skorin-Kapov 2013). Principally, measured factors which do not meet user and business requirements (e.g. SLA) are often identified as indicators of resources to be optimised with respect to technology and resource availability (Batteram et al. 2014). Generally, there are two types of QoE optimisation indicators that need to be determined and mapped in order to control QoE (Baraković & Skorin-Kapov 2013): Key Performance Indicators (KPIs) and Key Quality Indicators (KQIs).

- **KPIs:** are internal indicators derived from the measurements of network resources (e.g. objective measurements of performance, availability, reliability, usability).

- **KQIs:** are external indicators derived from KPIs and associated with different quality aspects that reflect user experience (e.g. subjective measurements of performance, availability, reliability, usability).

The mapping between KPIs and KQIs generates Customer Experience Indicators (CEIs), which ultimately provides objective measurements of customer experience (Stojanovic et al. 2015). In order to perform the mapping, QoS metrics (which ultimately become KPIs) and the level of customer satisfaction (which ultimately become KQIs) need to be defined, associated, and agreed in the SLA. Subsequently, both the KPIs and KQIs can be verified to meet SLA requirements and optimised, if necessary (Morais & Cavalli 2012; Stojanovic et al. 2015).

In terms of Web Application browsing, Batteram et al. (2014) categorised QoE indicators into three classes: Service availability' service responsiveness; and service quality, each of which is derived from the KQIs that are associated with the KPIs, i.e. KPIs, such as the session setup success ratio and end-to-end delay. KQIs, such as user satisfaction about the service downtime and response time between request and response. To emphasise the relative importance of each factor, the KQIs are weighted according to their importance and the criteria defined by service provider. Ultimately, the KQIs are grouped to represent the overall QoE indicators, which are used for system controlling and optimisation.

- **QoE Optimisation Points/Edges**

Along with the identified QoE indicators, the optimisation points and associated resources need to be identified accordingly to orchestrate and integrate the monitoring and controlling processes. Essentially, QoE can be optimised at various locations within the network, depending on the business and technological requirements (Ivesic et al. 2014; Tran et al. 2014). For example, in the context of Web Service, performance factor can be optimised by adjusting the configurations of the database production parameters or middleware parameters, depending on the resources available and the cost of any potential actions.

Generally, the optimisation points are the same as monitoring points, as in both cases the same metrics require capturing, but for different purposes. Regarding monitoring points, metrics are captured for measuring QoE, while, in relation to optimisation points, metrics are captured for controlling it (Baraković & Skorin-Kapov 2013). Optimising a specific QoE indicator can be performed at several optimisation points, depending on how the indicators of the KPIs and KQIs are associated with each other (Batteram et al. 2014). For instance, in the case where there are dependencies between the indicators, the optimisation process may initiate at one point and trigger at another (Baraković & Skorin-Kapov 2013). Furthermore, the optimisation process may be performed at different levels (i.e. layer-based solutions), from lower to upper layers. Lower layer solutions refer to physical infrastructure, network and data links. Upper layers solutions pertain to to application and the presentation layer. The combination of lower and upper layers is known as a cross-layer (Tran et al. 2014), which is most commonly prescribed as a solution for specifying QoE optimisation points as in Ameigeiras et al. (2010) and Ivesic et al. (2014). For each layer, the resources and their dependencies and measurement points are described, as well as the resources problem and solutions being specified (Tran et al. 2014).

- **The Invoking of the QoE Optimisation Process**

With regards to optimisation opportunities and timing when the QoE optimisation process should be initiated, several methods have been developed considering indicators related to the quality of the provided service (e.g. KPIs), and/or indicators related to user experience (e.g. KQIs). QoE optimisation can be conducted on an event-basis, i.e. when QoS is degraded or the level of churn is significantly high. In addition, QoE optimisation can be conducted on a regular-basis, where monitored parameters are evaluated and utilised along with the SLA, i.e. the optimisation process can be initiated automatically and dynamically in a unit time to adopt SLA (Morais & Cavalli 2012; Liotou et al. 2015).

Broadly, there are two methods for monitoring QoE and initiating the QoE optimisation process: In-service and out-of-service (Baraković & Skorin-Kapov 2013). The former is performed in an on-line mode and invokes the optimisation process dynamically towards the monitored indicators when the system is in operation. In-service is an important method of QoE management since QoS varies from time to time and also, satisfaction is reconceptualised (Takahashi 2009; Kim et al. 2010). Whilst the out-of-service method is performed in an off-line mode and invokes the optimisation process on request.

- **QoE Optimisation Mechanisms**

There are many mechanisms and approaches that have been proposed for monitoring and optimising QoE, thereby addressing the conflict of interests between the end-user and service provider. From an end-user perspective, a high quality standard of service is expected, whereas from the service provider perspective, it is important to provide a reliable service with relatively low-cost of deployment (Qadir et al. 2015). In other words, there needs to be adjustment of the balance between objective and subjective QoE factors (Wu 2010). There are several strategies that can be adopted for optimising and controlling QoE, which can be taken as being closer to the service provider or closer to the end-user. However, combining different approaches, will enable the optimisation process to be undertaken from different perspectives (Baraković & Skorin-Kapov 2013).

A survey conducted by Qadir et al. (2015) identified four mechanisms based on QoS and QoE functions for QoE optimisation, including: (1) QoE optimisation through rate adaption, which is mainly proposed for video traffic using a neural network to learn QoE scores and technical parameters and thereafter determining the optimal values; (2) QoE optimisation through cross-layer design, which uses prediction to promote dynamic feedback of end-to-end QoE for utilising resources and achieving high quality (Qadir et al. 2014); (3) optimisation through scheduling, which focuses on the delay as a distortion factor for packet loss ratio; and (4) QoE optimisation through content and resource management, which focuses on cache management for HTTP over the network.

Focusing on resource allocation and utilisation, Ivesic et al. (2011) developed a simulation tool that demonstrates resource allocation of the user's sessions and dynamically achieves optimisation with respect to the available resources. The tool is based on the Multi-choice Multidimensional Knapsack Problem (MMKP) approach for combinatorial optimisation problems, i.e. in contrast to the traditional approaches which focus on a single flow (e.g. packet delay), MMKP can deal with multiple flows (e.g. packet delay budget, packet error loss rate etc.). Beside the knapsack problem, a multi-objective optimisation problem (which is known as an NP-complete problem) can be generated for each flow to optimise the objective and subjective QoE values (ATNET 2011). Accordingly, both KPIs and KQIs can be formulated as a multi-objective optimisation problem. This is actually adopted for this research (in Chapter 6) as an approach for optimising and controlling QoE.

Table 2-6 summarises the QoE optimisation and control approaches discussed above, with respect to the following key elements: KPIs and KQIs, optimisation points, optimisation timing and initiatives, methods and mechanisms. A cross-layer approach is commonly used as a mechanism for performing QoE optimisation and allocation of resources (Wamser et al. 2013; Baraković & Skorin-Kapov 2013). In contrast, a few attempts have been made, such as in Ivesic et al. (2011) to bridge the gap between the KPIs and KQIs as well as to adjust the balance between user requirements and the available resources. This research considers the balance between the KPIs and KQIs and consequently, develops artefacts to optimise QoE in Chapter 7.

Table 2-6: Summary of QoE Optimisation and Controlling Models

Optimisation Approach	**QoE Optimisation Aspects**				**Target**		**Approach and Method Applied**	**Application Domain**
	Indicators	**Optimisation Points**	**Initiatives**	**Methods**	**User Satisfaction**	**Quality & Resources**		
(Batteram et al. 2014)	√	√	x	√	√	√	From network management to service quality management	NGN Applications and Multimedia Networks
(Wamser et al. 2013)	√	√	x	√	√	√	Cross-layer resource management	YouTube Video Streaming Services
(Ivesic et al. 2014)	√	√	√	√	√	√	Cross-layer QoE-driven	Multimedia and Network Services
(Tran et al. 2014)	√	√	√	√	√	√	A layer-based classification	Heterogeneous Wireless Network
(Liotou et al. 2015)	√	√	√	√	√	√	End-to-end model for QoE provisioning	Mobile Cellular Networks
(Ivesic et al. 2011)	√	x	x	√	x	√	Multi-choice multidimensional knapsack problem	Adaptive Multimedia Services
(De Vleeschauwer et al. 2008)	√	√	x	x	x	√	Overlay network deployment for end-to-end QoE	Network Routing Services
(Stojanovic et al. 2015)	√	√	√	√	√	√	Layered approach (infrastructure, performance, and the user awareness)	Internet Quality / Future Internet

2.4 Challenges in Adapting QoE to Web Application

Whilst, in particular, the QoE approach was introduced for multimedia and ICT services and multimedia applications, it has been extended for web services (Nguyen et al. 2013; ITU-T 2014; Skorin-kapov & Barakovic 2015; Yamauchi et al. 2015) as well as for cloud applications (Hobfeld et al. 2012). Furthermore, other researchers have drawn attention to loosely-coupled and interoperable services, such as Zieliński et al. (2012), who have proposed an adaptive SOA solution that integrates SOA with QoS and QoE. However, most of these studies addressed QoE of Web Applications from an ICT-oriented point of view and failed to assess it in the form of a multi-tiered application architecture, formed from the interfaces between business, presentation, business logic, data, middleware and network. This limitation leads to the following challenges.

- First, according to the studies summarised in Table 2-1, it is found that the lack of formulating the web objective metrics from the components of Web Application architecture leads to the definition of naive and substandard metrics that may not meet with web quality requirements, which specified and discussed in Appendix A (e.g. ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2012).

- Second, according to the studies that have been discussed and summarised in Table 2-1, it is found that having substandard objective metrics leads to the defining of inconsistent subjective metrics that do not align with the Web Application architecture nor web quality requirements, i.e. in order to evaluate end-user satisfaction on a Web Application or service, the subjective metrics (e.g. questions) need to be designed in a way that associates and aligns them with web architecture.

- Third, according to the studies summarised in Table 2-3 and Table 2-4, it is found that the QoE assessment process is a task that fluctuates from business to business and from technology to technology, i.e. monitoring can be performed in different ways depending on the scope and purpose (Al-Moayed & Hollunder 2010). Since QoE objective factors are generally expressed by a set of technology-platform dependent parameters and their relative subjective factors are associated and mapped accordingly, QoE factors (including objective and subjective factors) need to be derived in accordance to the characteristics of web technology, i.e. ISO/IEC TR 9126-3 2002.

- Fourth, due to the general problem of quantifying QoE in most application domains, which is addressed in Section 2.3.4, in the same sense, quantifying QoE in Web Application is still a challenge, i.e. due to the lack of methods or mechanisms that define the mapping and relationship between the objective and subjective factors(Alreshoodi & Woods 2013; Aroussi & Mellouk 2014; Fiedler et al. 2010; Schatz et al. 2013; Laghari & Connelly 2012)

- Fifth, in terms of predicting and perceiving QoE, it is found in Section 2.3.4 that the performance of ML classifier is based on the training set derived from the application domain (Alreshoodi & Woods 2013). Consequently, it is important to understand the nature of objective and subjective factors and how they correlate with each other, i.e. understanding the correlation between Web Application quality and user experience helps to obtain better prediction performance.

- Sixth, and finally, in order to invoke a QoE optimisation process and to utilise resources, it is necessary to understand the characteristics of these resources including the environment, business and technology. In addition, based on the discussion that have been summarised in Table 2-6, it is found that it is important to know how resources are associated with the objective and subjective indicators (KPIs and KQIs), i.e. understanding the characteristics of Web Application resource helps to improve optimisation.

2.5 Summary

This chapter has provided the overall context for this study. It can be summarised that Web Application technology has become an essential element of ICT services, especially for commercial enterprises. With this evolution, the demand for high quality of Web Application has increased considerably and it has become an important aspect in web engineering. However, in order to evaluate the quality of a Web Application effectively and to investigate whether its intrinsic limitation is related to objective or subjective aspects or to both, it is important to understand its architecture (Lew et al. 2012). This is because this plays a vital role in determining quality attributes and sub-attributes of the components of the architecture (Babar 2008). In term of Web Application quality, most of the available models are based on quality factors that are derived from ISO-9126, ISO -9241 and OASIS, which focus on the quality of the service from the service provider's point of view, thereby neglecting the quality of the user's experience, which is subjectively measured.

Consequently, this study involves investigating a QoE approach, which aims to measure the overall service quality as perceived by customers. It is found that QoE management requires four main processes to achieve the maturity of QoE assessment: (1) QoE modelling process; (2) QoE monitoring process; (3) QoE measuring process; (4) and QoE controlling and optimising process. Finally, this chapter has discussed the challenges in utilising QoE in Web Application. It is concluded that the following challenges are considered as general fundamental problems for the process of QoE assessment in most application domains: (1) The quantification of QoE; (2) the prediction of QoE; (3) the optimisation of QoE; (4) and the user perception of QoE. These challenges lead to address the problem that this research is intended to solve.

Chapter 3: Research Design and Approach

3.1 Overview

This chapter discusses the research methodology adopted for this research project and presents how it is applied. This chapter is structured as follows: Section 3.2 presents an overview of research approaches in IS, focusing on the behaviour-science and design-science paradigms. Section 3.3 discusses the philosophical grounding of the DSR approach. Section 3.4 presents how DSR is applied in this research, describing the awareness of the problem, the suggested solution, the development of the artefacts and the evaluation process. Section 3.5 provides a conceptual view of the development of the proposed model (QoEWA) and Section 3.6 presents a summary of this chapter.

3.2 Research Approaches in IS

Several research paradigms have been proposed in IS research that together develop interdisciplinary perspectives on problem-solving environments. For instance, some focus on positivist studies, which generally involve adopting a deductive approach, where the researchers are independent from the study and work within a theoretical perspective (Dubé & Paré 2003). Whilst others focus on the interpretivist studies, which are invariably conducted by researchers who are more concerned with understanding subjective experience and the sense behind actions in social contexts (Lin 1998). However, due to the multidisciplinary nature of IS, it is important not to remain restrictive with a single research methodology, for research in this domain is generally pursued using a multi-paradigmatic approach based on the behavioural-science and design-science paradigms (Orlikowski & Baroudi 1991; Hevner et al. 2004).

The behavioural science paradigm has its origins in natural science research and it was derived from various behavioural disciplines with the aim being to understand organisational and human phenomena, by integrating the analysis, design, implementation and management through information systems (Hevner et al. 2004). Behavioural science research focuses on the development and justification of theories that describe phenomena related to organisational needs, i.e. principles and laws. Moreover, it produces theories that seek to find truth by explaining and predicting behaviour of individuals or organisations (Hevner & March 2003).

The design-science paradigm has its origins in engineering and artificial science. It is a problem-solving paradigm, which focuses on the building and evaluation of artefacts developed for identified business needs, such that these can produce results with reference to behavioural science (March & Smith 1995). Design science seeks to use information systems efficiently and to promote innovations as well as deliver artefacts that represent the practices, technical capabilities and products needed for accomplishing analysis and design. Generally, design science research in information systems has four types of artefacts (Hevner & March 2003):

- Constructs: Providing languages for defining problems and solutions.
- Models: Represent the connections between problems and solutions.
- Methods: For defining solutions processes.
- Instantiations: Describe how to implement constructs, models and methods.

Whilst the behavioural-science and design-science paradigms are fundamentally different, they do depend on each other. There is a complementary nature between behavioural-science and design-science research that addresses the fundamental problem in the application of information technology (Hevner & Chatterjee 2010). According to Hevner et al. (2004), the object of study in behavioural science research in the information systems field is normally an information technology artefact that is already implemented in an organisational context and created through IS design science research. The design process and the evaluation of the artefacts are iterated through a design science research cycle, which includes rigour and relevance cycles.

For understanding, executing, and evaluating IS research, Hevner et al. (2004) presented a conceptual framework, combining the behavioural-science and design-science paradigms, as presented in Figure 3-1. The framework describes the process of developing the artefact, which is derived from the knowledge-base and environment. The former consists of foundations (e.g. theories and methods) and methodologies (e.g. formalisms and techniques) that provide knowledge and information about the planned research to be conducted in a particular environment, where the problem space is already defined. The environment is composed of people, organisation (e.g. business strategies and structure) and technology (e.g. applications and infrastructure). The relation between the knowledge base and environment is established through IS research, which is addressed by behavioural-science (e.g. develop and justify theory) and design-science (e.g. build and evaluate theory).

The utility and quality of a developed artefact must be rigorously implemented, which describes the way in which study is conducted. In behavioural-science, the assessment of rigour is generally based on appropriate data collection and analysis methods, but in design-science it is based on the researcher's skill (Hevner & Chatterjee 2010) and can be assessed by computational approaches (e.g. mathematical formalisms and computer simulation) that describe the constructed artefact. In IS research, the relevance phase initiates the research with an application context that addresses the research problem and identifies possible solutions (Hevner et al. 2004) and (Hevner & March 2003).

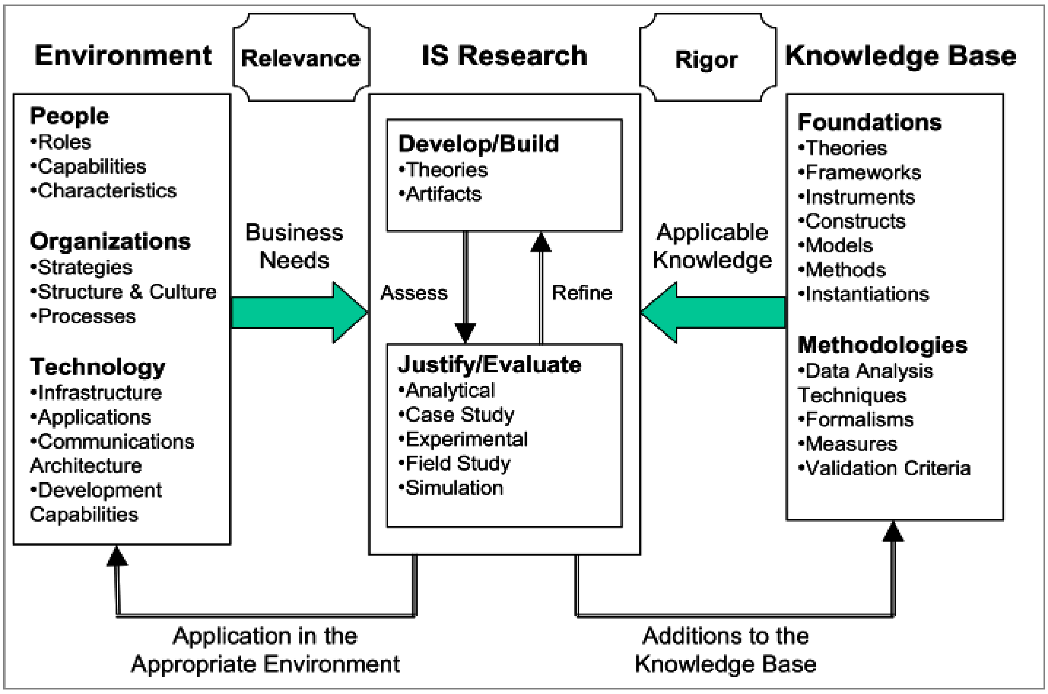


Figure 3-1: IS Research Framework Source: (Hevner et al. 2004)

Building upon the above, behaviour-science and user participation can be integrated into the design-science process in IS research (LeRouge & Lisetti 2006). This combination is taken as a new research direction bounded by design science research (DSR) approach, which is discussed in the next section and eventually utilised as an approach for this research project.

3.3 Design Science Research (DSR) Approach

Whilst there has been much discussion on DSR in the literature to-date, there is little in the way of work that provides evidence of the interaction between problem and solution spaces, thus showing how design theory is grounded in practice. In addressing this issue, this section highlights the key points that frame the use and subsequent discussion of DSR: Design theory, the importance of iteration and the creativity inherent in the process.

Broadly speaking, theory in DSR has been discussed in terms of informing the design, as a means of expressing design knowledge and as an outcome of the design instantiation. Design may be informed by kernel theory, generally taken as the underlying knowledge or theory imported from other fields of interest that provide a basis and/or explanation of aspects of the design (Walls et al. 1992; Jones et al. 2007; Kuechler & Vaishnavi 2008).

Importantly, however, some have argued that focusing on kernel theory is a potential distraction to artefact design itself (Orlikowski & Baroudi 1991). Theory has also been considered as a means by which design knowledge is captured, formalised and communicated, consequently possibly taking a different form to other disciplines (Walls et al. 1992; Jones et al. 2007).

As an outcome, theory can contribute to research and practice bi-dimensionally through originality and utility (Gay & Weaver 2011). In squaring the circle here, one perspective on this is that kernel theories can be refined and developed by DSR as an outcome of the design (Kuechler & Vaishnavi 2008), thereby contributing to a theory's explanatory power or incrementally adding to the lexicon of facts, for example. More pragmatically, perhaps, Venable (2006) proposes utility theory as a (generalisable) mapping between problem and solution space, suggesting the following prototypical forms:

- (New) Technology X (when applied properly) will help in effectively solving problems of type Y;
- (New) Technology X (when applied properly) will efficiently provide improvements to type Y;
- (New) Technology X (when applied properly to problems of type Y) is more effective than technology Z.

The above follow the generally accepted view that DSR addresses unsolved problems in unique or innovative ways or solves problems in more effective or efficient ways (Hevner et al. 2004). In doing so, the naïve view of design as a rational and linear process that moves from problem to solution via a set of fixed moves (representing theories, methods, heuristics etc.) should not form the basis of presentation. Positively, DSR as a means of learning via the act of building, is one area where there is consensus in the literature (Kuechler & Vaishnavi 2008). In this research, the observation, however, is that this type of learning is not well-evidenced in published work to-date; design decisions often remain opaque as do the iterative/incremental steps in the design process (even though software development methods have evolved to address them explicitly). If design theory is taken according to how Jones & Gregor (2007) see it, then more explicit consideration may be warranted. This is of particular salience if one accepts the position that the creative aspect in design is not a sudden 'leap', but rather, emerges as a (temporary) bridge from the co-evolution of problem and solution spaces during the design process (Dorst & Cross 2001). The design process is not linear (Gregor 2009) and work in the solution space often reframes the problem space. If so, (a) design theory is more 'grounded' in practice in a way that it should be acknowledged and (b) iterative and/or incremental learning forms an important part of that theory.

Accordingly, the practical work of this research takes the form of four design-build-evaluate iterations. This researcher remains mindful of popular process models and guidelines for DSR (Hevner et al. 2004; Kuechler & Vaishnavi 2008; Peffers et al. 2008), but uses the more generic form proposed by Walls et al. (1992) for discursive ease regarding the iterative aspects of the work. This approach also allows for a simplified mapping of the work with the skeleton of a design theory covering (Jones et al. 2007): (a) The purpose and scope of the theory; (b) constructs; (c) the principles of form and function; (c) artefact mutability; (d) testable propositions; (e) justificatory knowledge (kernel theory); (f) principles of implementation; and (g) each expository instantiation. This approach is illustrated at Figure 3-2

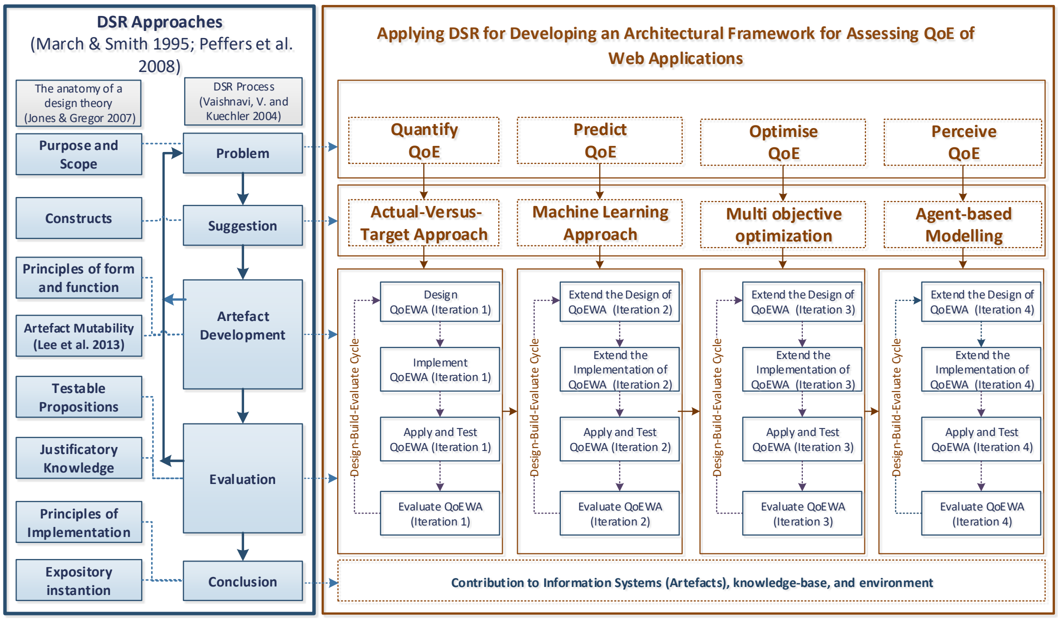


Figure 3-2: Research approach and process adopted

3.4 The Application of DSR

Due to the multidisciplinary nature of the QoE approach discussed in Chapter 2 (Laghari & Connelly 2012; Geerts et al. 2010), the assessment of QoE is generally performed by a combination of objective and subjective factors (Soldani et al. 2006; Mitra et al. 2011) that somehow are associated with aspects of human behaviour and design science (Laghari & Connelly 2012). Consequently, the DSR approach is applied in this research to study the characteristics of the objective and subjective factors, as well as the relationship between them. Also, DSR is considered here as a methodology to develop the research through guidelines and stages that are framed by models, such as those of Hevner et al. (2004) and Vaishnavi and Kuechler (2004). The stages include: (1) Awareness of the problem; (2) solutions and suggestions; (3) development; (4) evaluation, (5); and (6) conclusion and outcomes. The primarily focus here is on the production of a DSR artefact, which has utility clearly associated with potential practice.

3.4.1 Awareness of the Problem

As discussed in the literature review in Chapter2, the QoE approach was originally introduced for multimedia and network services (Laghari & Connelly 2012; Geerts et al. 2010), but it was subsequently been extended for web services (Nguyen et al. 2013; ITU-T 2014; Skorin-kapov & Barakovic 2015; Yamauchi et al. 2015) and Cloud applications (Hobfeld et al. 2012; Cecchet et al. 2013). Other studies drew attention to loosely-coupled and interoperable services, such as the model proposed by Zieliński et al. (2012), which pertains to an adaptive SOA solution that integrates SOA with QoS and QoE.

However, in the context of web services, it is noted that most QoE studies have not paid attention to the aspect of IS, which provides the knowledgebase and methodologies required for web development (Hevner et al. 2004). Moreover, there is still a lack of rigour in defining the QoE factors as most of current QoE models are based on ITU factors (ITU-T 2006b; ITU-T 2014), which are extracted from network and multimedia domains, as illustrated in Figure 3-3.

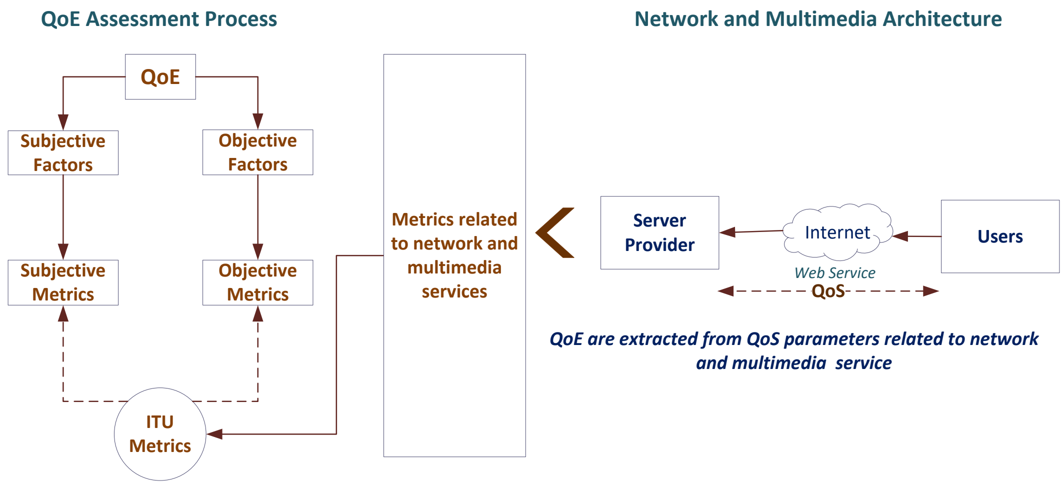


Figure 3-3: Traditional process of extracting QoE factors

The absence of formulating the web QoE factors from the core of the web application architecture leads to the identification of naive and substandard factors and metrics that may not meet web and software quality requirements, which are generally specified and standardised by models such as: (ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2012). That is, having substandard objective metrics leads to the definition of inconsistent subjective metrics that do not align with the Web Application architecture, as well as the web quality characteristics. Alongside these problems, it was found in the literature review conducted in Chapter 2 that the following challenges are considered as fundamental problems for QoE assessment (Skorin-kapov 2012; Hobfeld et al. 2012; Baraković & Skorin-Kapov 2013). As aforementioned in Chapter 1, the four key challenges focused on in the current research are:

1. The quantification of QoE.
2. The prediction of QoE.
3. The optimisation of QoE.
4. The perception of QoE.

3.4.2 Suggested Solution

The solution suggested in this research goes beyond the traditional QoE approaches, which assess it in the context of multimedia and network services. For here, a novel model is proposed called QoEWA (QoEWA is an acronym standing for QoE of Web Application) that is deemed particularly appropriate for assessing the QoE of Web Applications. The model differs from others in that it assesses QoE in terms of Information Systems and with consideration of the dimensions of web quality requirements and web multi-tier architecture.

Figure 3-4 illustrates the architecture of QoEWA and shows how the web QoE factors are extracted from the main components of the web architecture in accordance with the ISO-9241 and OASIS quality models (ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2012)

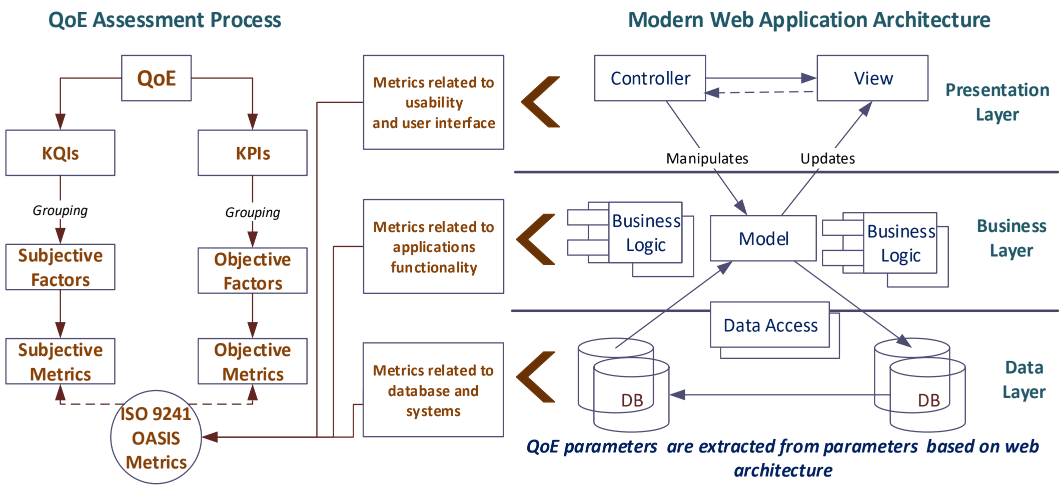


Figure 3-4: The proposed process of extracting QoE factors

In addition to the proposed the QoE architecture illustrated above, the suggested solution also considers the QoE assessment aspects defined earlier in Subsection 3.4.1. Accordingly, the solution also aims to develop artefacts that enable the proposed model to achieve the following.

- **Quantify QoE:** For this research, an Actual-Versus-Target approach based on the correlation between the KPIs and KQIs (Kan et al. 2001) is adopted so as to bridge the gap between the actual measurements and those defined by the Service Level Agreement (SLA). The ratio between actual and target measurements allows the model to compute the Actual-Versus-Target area, which is ultimately used for quantifying QoE.

- **Predict QoE:** Machine Learning (ML) is drawn upon to measure intelligently the subjective factors, which are typically measured by Mean Opinion Score (MOS) tests to assess how service quality is perceived by customers (Khan et al. 2012), i.e. the correlation between the KPIs and KQIs enables the model to predict the unknown KQIs from the known KPIs (Alreshoodi & Woods 2013).

- **Optimise QoE**: This research involves adopting a Multi-Objective Optimisation (MOO) approach to determine the optimal QoE value, which maximises the perceived user experience and minimises the technical resources (Baraković & Skorin-Kapov 2013), i.e. QoE is optimised when the service is provided with minimal technical resources at an acceptable level of satisfaction (Agboma & Liotta 2008).

- **Perceive QoE**: An Agent-based Model (ABM) system that examines and simulates the user's behaviour and interaction is developed. This enables the decision maker to understand the factors behind QoE, and it adjust the value of the KPIs in a way that improves the KQIs as well as increasing the number of promotors, whilst decreasing detractors.

3.4.3 Development of QoE of Web Application (QoEWA)

The suggested solution presented in the previous section is developed at this stage. QoEWA is developed iteratively, whereby each developed iteration includes a set of artefacts that are evaluated based on a build-and-evaluate cycle that provides feedback and a better understanding of the utilised solution. Consequently, in the following build-and-evaluate process, the development of each iteration in this research is structured into four sections as follows: (1) a build and design section; (2) an instantiation and implementation section; (3) an application and testing section; and (4) an evaluation section. Since the artefacts are generally classified into four types, namely, construct, model, method, and instantiation (Hevner & March 2003), the build and design section includes construct, model, and method artefacts. Whilst the instantiation and implementation section includes instantiation artefacts. The key artefacts are developed iteratively through four iterations:

- Iteration 1: Utilises the Actual-Versus-Target approach to enable QoEWA to quantify QoE (developed in Chapter 4);

- Iteration 2: Utilises a Machine Learning (ML) approach to enable QoEWA to predict QoE (developed in Chapter 5);

- Iteration 3: Utilises the Multi-Objective Optimisation (MOO) approach to optimise QoE (developed in Chapter 6);
- Iteration 4: Utilises the Agent-Based Modelling (ABM) approach to perceive of and gain insight into QoE (developed in Chapter 7).

3.4.4 Evaluation Stage

This is the stage in DSR where the developed artefacts are tested by comparing performance to the criteria specified in the suggestion stage. The evaluation process provides feedback and a better understanding of the identified problem. The build-and-evaluate cycle improves the quality of the suggested solution, which is iterated and repeated a number of times before the final version of the artefact is produced. The evaluation of the designed artefacts is generally based on the method provided in the knowledge-base, where the selection of the method is matched with the designed artefact and the evaluation metrics (Hevner et al. 2004). The evaluation strategy of each developed iteration is outlined as follows.

- **Evaluation of Iteration1:** It is conducted statistically over two tests. The first test provides a benchmark in relation to the state-of-art, with the correlation between the KPIs and KQIs being examined based on the assumption, confirmed by the fact, that a strong positive correlation indicates an excellent relationship between the objective and subjective factors (Upadhyaya et al. 2014). While the second test examines the efficiency of QoEWA in quantifying QoE. The obtained quantified value in the second test is considered to be consistently reliable, when the correlation between the KPIs and KQIs from the first test is positive and significant (Qianqian Xu et al. 2010).
- **Evaluation of Iteration 2:** It is conducted by a test that statistically examines the scenario (based on ML evaluation methods), where QoEWA is trained on data obtained from previous feedback about the service quality, thereby

providing a comparative assessment of the ML algorithms for predicting the poll scores. The test examines how well ML algorithms (DT, NB, SMO, IBK and RF) can perform the QoE prediction process. The efficiency of each is evaluated by the following tests: True Positive (TP), False Positive (FP), Precision, Recall, F-measure, and the Operating Characteristic model (ROC) (Witten et al. 2011).

- **Evaluation of Iteration 3:** It is conducted by two tests based on a simulation. The first, is designed to validate the model by exploring the data and comparing it with the outcomes obtained in iteration 1. The second test is to compute and predict the optimal values of QoE. The values obtained from the second test are compared with the real values calculated in Iteration 1. The assumption made here is that the best optimal values calculated by the simulator in the second test have be defined within the range calculated by the first test for the best optimal values, which are based on a real dataset.
- **Evaluation of Iteration 4:** The test is rigorously verified and validated through the guidelines proposed by Rand & Rust (2011) and the taxonomy of the agent-based model presented by Windrum et al. (2007). This combination allows each test case to be independently performed under consideration of several factors, including: (1) The nature of the object under test; (2) sensitivity analysis with respect to the obtained optimal values; and (3) verification and validation. The developed ABM system is tested in three scenarios. The first presents the process which initiates and explores the current situation. The second presents a what-if scenario and examines agent's behaviour, whilst the third examines agents' behaviour and interaction. The assumption here is that the results in the first test must meet the values measured in Iteration 1, which as previously mentioned is based on a real-life dataset. Moreover, the results of the second and third tests must be within the range identified in the first test.

3.4.5 Outcomes and Conclusion Stage

It is the final process of the research cycle where the results and outcomes are presented. In DSR, the conclusion includes knowledge and experience which are obtained through this cycle. This knowledge can be summed up as a list of guidance for practitioners who are intending to use or apply the developed artefacts. To assess the research contribution, there are seven guidelines proposed by Hevner et al. (2004) that can be used for conducting and presenting design science research.

3.5 Setting the scene of the development of QoEWA

The purpose of this section is to provide a conceptual view of the development of the QoEWA model and its main elements; mapping the problem domain with the solution domain. This basically gives the reader an introduction and a coherent view regarding the techniques and approaches utilised in each iteration.

3.5.1 QoEWA Model

In moving from problem space to solution space, key design decisions are required in relation to the constructs that define and map the so called ‘objective’ and ‘subjective’ factors. Consequently, this section defines a set of quality factors (F1,F2,F3,F4, and F5) derived from standard models, such as ISO 9241-11 (1998), ISO/IEC TR 9126-3 (2002) and OASIS (2012) as well as a set of usability factors (F6,F7,F8, and F9) derived from models, such as those of Seffah et al. (2006), Mifsud (2015) and Hussain and Kutar (2009). In addition, other non-technical factors, such as (F10 and F11) are derived from QoE ecosystem models (Laghari & Connelly 2012; Skorin-kapov 2012). This is because some researchers argue that the quality of Web Application can be affected by technical and non-technical aspects (Negash et al. 2003; Udo et al. 2010), e.g. governance, customer services and/or IT support. Since this research focuses on QoE, as mentioned earlier, the review of the quality models and their factors is outside of its scope. However, detailed description of the quality models and their factors are discussed in Appendix A. Each factor has objective and subjective formulae, as shown in Table 3-1. The input of the former is extracted from the operational data sources (e.g. applications, middleware and database), whereas that for the latter is taken from the MOS test, which is stored in the Customer Relations Management (CRM) system.

Table 3-1: List of the objective and subjective factors

Reference	**Factors**	**Objective Metrics** (ISO 9241-11 1998; ISO/IEC TR 9126-3 2002; OASIS 2012)	**Objective Formula**	**Subjective Metrics** *(formulated subjectively based on the objective metrics defined by the ISO and OASIS quality models)*
F1	Performance	m1: Maximum completed requests m2: Unit time	$F1 = (m1/m2)$ (max throughput)	Are users satisfied with the time taken to send a request and receive a response from their terminals or web page?
F2	Reliability	m3: Number of correctly implemented items m4: Total number of compliance items	$F2 = (m3/m4)$ (compliance)	Are users satisfied with the number of successful performed tasks over a period of time?
F3	Availability	m5: Down-time m6: Unit-time	$F3 = 1 - (m5/m6)$	Are users satisfied with the availability of the application and the operational uptime?
F4	Accessibility	m7: Number of acknowledgement messages m8: Number of request messages	$F4 = (m7/m8)$	Are users satisfied with the ratio of the successfully returned acknowledgements

				after requesting tasks?
F5	Success-ability	m9: Number of responses m10: Number of requests	$F5 = (m9/_{m10})$	Are users satisfied with the ratio of the requests (sent by user) to responses (performed by server provider)?
F6	Learnability	m11: Number of functions described m12: Total number of functions provided	$F6 = (m11/_{m12})$	Are users satisfied with the simplicity and the functions implemented with help facility and/or documentation?
F7	Operability	m13: Number of instances of operations with inconsistent behaviour m14: Total number of operations	$F7 = 1 - (m13/_{m14})$ (operational consistency)	Are users satisfied with the number of operations (e.g. form layout) with consistent behaviour?
F8	Usability (Effectiveness)	m15: Number of tasks completed successfully m16: Total number of tasks	$F8 = (m15/_{m16})$ (completion rate)	Are users satisfied with number of tasks completed successfully in a given time?
F9	Usability (Efficiency)	m17: Number of correctly implemented items related to efficiency compliance confirmed in evaluation m18: Total number of compliance items	$F9 = (m17/_{m18})$ (compliance rate)	Are users satisfied with the time taken to complete a number of tasks from their terminals in accordance with the compliance requirements?
F10	Responding to users	m19: Time taken to respond to user m20: Maximum time to respond as in SLA	$F10 = (m19/_{m20})$	Are users satisfied with the time taken to receive a response from customer support?
F11	Professionalism	m21: Time taken to fix issue m22: Maximum time to fix issues as in SLA	$F11 = (m21/_{m22})$	Are users satisfied with the quality of the technical support received from customer services?

Figure 3-5 describes the process of computing the objective and subjective factors and illustrates how they are correlated and mapped to quantify and predict QoE. QoEWA includes three core processes that are used in each iteration as a routine procedure for assessing QoE.

- **KPIs assessment process:** Is a task that fluctuates from business to business and from technology to technology, depending on the scope and purpose of the service (Al-Moayed & Hollunder 2010). In this research, this process is based on the objective formulas, which have been defined earlier in Table 3-1 to compute the KPIs and is described as:

  KPIs = {F1obj, F2obj, F3obj, F4obj, F5obj, F6obj, F7obj, F8obj, F9obj, F10obj, F11obj}, where 'obj' indicates objective factor **(1)**

- **KQIs assessment process:** Is a task that assesses the subjective factors and is associated with the KPIs assessment process for computing the KQIs. It is commonly performed by an MOS test (ITU-T 2006b), which is based on an ordinal scale of five-points: (1) bad; (2) poor; (3) fair; (4) good; (5) excellent. A KQI is described as:

  KQIs = {F1sub, F2sub, F3sub, F4sub, F5sub, F6sub, F7sub, F8sub, F9sub, F10sub, F11sub}, where 'sub' indicates subjective factor **(2)**

- **Mapping process:** Since the mapping between the objective and subjective factors is an essential and important task in the life-cycle of a QoE assessment (Hobfeld et al. 2012) as it defines the relationship between the objective and subjective metrics (Fiedler et al. 2010), complementary approaches of correlation analysis and two-dimensional gap analysis that maps between the KPIs and KQIs is adopted. The mapping is a task that essentially combines a set of relatively objective metrics into a single KPI and thereafter, associates that KPI with a specific one expressed by a set of subjective metrics (i.e. these are usually expressed as a question). This mapping is expressed as:

QoE = {KPIs, KQIs}, where the point (KPIs and KQIs) are represented as nominal values. **(3)**

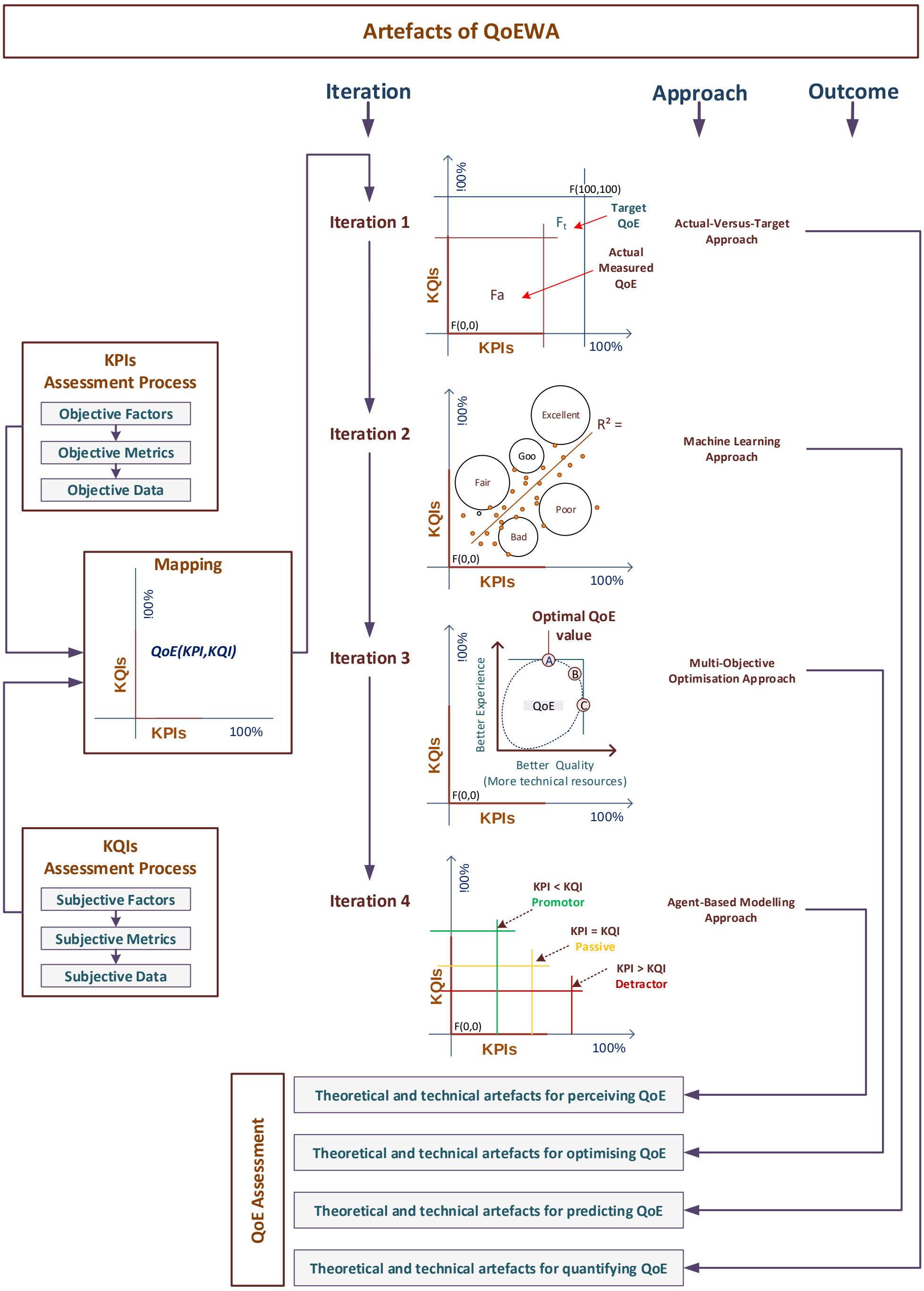


Figure 3-5: Conceptual design structure of QoEWA

As a result of the mapping, KPIs and KQIs are correlated to determine the degree of association between the objective and subjective aspects of QoE. This facilitates the development of the artefacts, which are iteratively constructed through three iterations framed by DSR methodology. Figure 3-5 describes how the results obtained from the correlation of KPIs and KQIs feed the iterations, which are developed one after the other to refine the design of QoEWA and the quality of QoE assessment.

3.5.2 Case Study Description

The work is undertaken in the context of Web Applications employed within a UK University, whereby continuing issues around service quality, led to a continuous improvement solution to this being sought.

The resulting dataset has nearly 100,000 sessions collected over a 12 month period from 335 users, who used four 4 different web-based applications installed across two web servers. Figure 3-6 describes the sources of the KPIs and KQIs datasets i.e. KPI data encompasses the objective data, while the KQI data encompasses the subjective data.

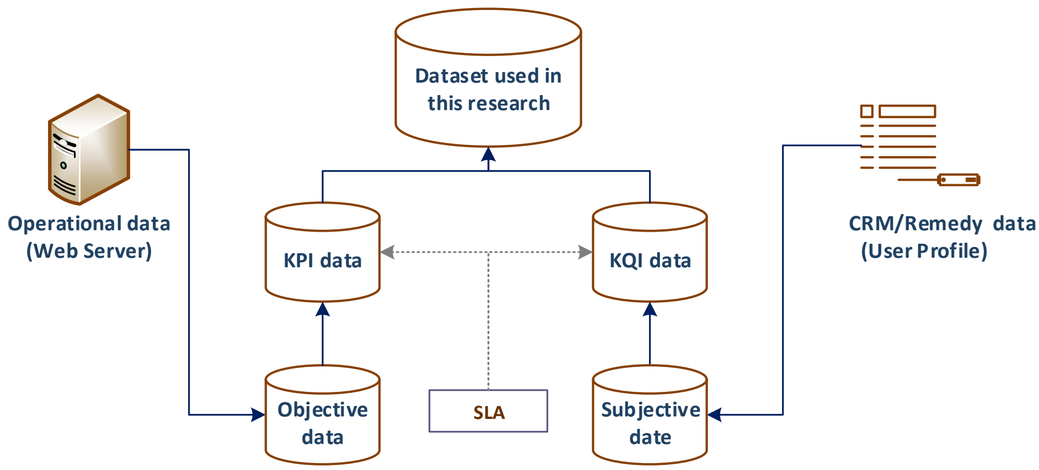


Figure 3-6: Sources of the dataset

Table 3-2 summarises the statistics of the dataset, showing the mean, standard deviation, maximum and minimum values of each objective and subjective factor.

The table shows eleven objective factors (KPIs) defined in association with the metrics used to capture the data, which relate to the technical and non-technical aspects of quality. Alongside these factors, there are eleven subjective ones (KQIs) extracted from user profiles to rate satisfaction, each of which is weighted, according to its importance, (Behkamal et al. 2009) by calculating the ratio of the total numbers of remedies serviced with high priority and the overall total numbers of those serviced with high priority for all factors. To keep consistency between the KPIs and KQIs, the scale of each factor is expressed as a percentage of the ratio of the difference between the measured value and the target value (Yetgin & Göçer 2015). Moreover, specifications of each factor (e.g. acceptable values, targets, priorities, user requirements and business requirements) are stored in an SLA, which reflects the quality and system requirements for both the KPIs and KQIs (TOGAF 2004).

Table 3-2: Summary of the KPI and KQI values obtained from the dataset

QoE Measurement		Factors										
		F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
Values obtained from Web Server												
KPIs	Mean	60.48	58.94	58.68	59.67	57.63	56.65	56.17	53.80	54.95	57.15	55.11
	Maximum	90.86	88.56	88.17	89.66	86.59	85.12	84.39	80.83	82.56	85.87	82.80
	Minimum	19.75	19.25	19.17	19.49	18.82	18.50	18.35	17.57	17.95	18.67	18.00
	Standard Deviation	22.61	22.03	21.94	22.31	21.54	21.18	21.00	20.11	20.54	21.36	20.60
Values obtained from user profiles												
KQIs	Mean	61.00	59.19	55.98	56.93	52.09	56.89	56.41	51.32	52.42	57.39	52.57
	Maximum	99.00	98.19	92.86	94.43	86.40	94.37	93.57	85.14	86.95	95.20	87.21
	Minimum	19.75	19.25	18.21	18.52	16.94	18.50	18.35	16.69	17.05	18.67	17.10
	Standard Deviation	23.00	22.79	21.55	21.92	20.06	21.90	21.72	19.76	20.18	22.10	20.24
Values obtained from SLA and Remedy												
Weight	Scale of the factor	%	%	%	%	%	%	%	%	%	%	%
	No of Remedies	17	16	17	13	15	16	13	13	13	14	13
	Weight of the factor	0.11	0.10	0.11	0.08	0.09	0.10	0.08	0.08	0.08	0.09	0.08

3.6 Summary

This chapter has presented an overview of IS research approaches and it has focused on the DSR approach adopted for this research project. This chapter has highlighted the key points that frame the use and subsequent discussion of DSR in relation to the design theory, the importance of iteration and the creativity inherent in the process. The application of DSR has been presented through five stages and guidelines formulated according to this paradigm. The first stage, is the initial activity of the DSR process aimed at briefly defining the research problem of QoE regarding a Web Application, and scoping this from an IS perspective. The output of this activity is a preliminary research proposal that addresses four challenges in the QoE of a Web Application, including: (1) quantification of QoE; (2) prediction of QoE; (3) optimisation of QoE; and (4) user's perception of QoE. The second stage evaluates potential solutions to the defined problem. Based on literature emerged in Chapter 4, 5, 6, and 7, there are four of these proposed approaches: (1) Actual-Versus-Target approach; (2) Machine Learning; (3) Multi-objective Optimisation; and (4) Agent-based Modelling. The third stage, involves implementing the produced tentative design through four iterations, each of which is constructed and developed in a separate chapter (Chapter 4, 5, 6, and 7). The fourth stage provides feedback and a better understanding of the identified problem, such that evaluation of the strategy for each iteration can be undertaken. The fifth and final stage, provides knowledge and experience, which are obtained through the research life cycle. The conclusion of the developed iterations is provided in Chapter 8. According to the outlined five stages, this chapter has provided a coherent framework that identifies the link between the problem, solution and the developed model as well as its evaluation.

Chapter 4: The Quantification of QoE

4.1 Overview

This chapter develops the initial design of QoEWA to model and quantify QoE. This chapter is structured as follows: Section 4.2 presents an overview of the Actual-Versus-Target approach that is adopted in this research to quantify QoE. Section 4.3 presents the development of Iteration 1 through four subsections that describe the design, instantiation, application and tests as well as evaluation. Section 4.4 presents a summary of this chapter.

4.2 Actual-Versus-Target Approach for Quantifying QoE

Actual-Versus-Target is a gap analysis technique that compares the difference between the current actual performance of an activity and its target performance. It is commonly used in the performance management and compliance monitoring process to ensure productivity and to verify compliance requirements (Eckerson 2009). In order to achieve efficient performance management, it is necessary to define a set of KPIs that measure the ratio of actual performance to its target, i.e. each KPI can have a target that indicates the value to be achieved. According to Eckerson (2009), there are six essential elements that must be considered when defining a KPI:

- **Strategy:** It embodies a strategic vision and determines performance against a goal;
- **Target:** It has to measure a specific target that is defined in the strategy;
- **Ranges:** It has a range of performance, e.g. above, on or below target;

- **Encoding:** It and its ranges are encoded, e.g. it can be based on a percentage;
- **Time frame:** It has a target that assigned to a time frame;
- **Benchmarks:** It has a target that is measured against a baseline and benchmark comparisons.

In performance management strategies, there are two fundamental types of KPIs: Outcome and driver. An outcome KPI pertains to a lagging indicator that measures the output of past activity. The driver KPI is referred to as a leading indicator that measures the current and future state of activity, as well as the progress towards achieving goal. The difference between the two is that the outcome is recognised as a strategic KPI that focuses on strategy and management and is generally used by executive users on monthly or quarterly bases. While the driver is recognised as an operational KPI that focuses on control operation and monitoring and is generally used by operational staff on a daily basis. Both are associated and complement each other (TOGAF 2004). Another distinction between KPIs is that some are derived from a quantitative data, while others are derived from qualitative data. A quantitative KPI is represented as an interval or ratio values, whereas a qualitative one is assigned nominal or ordinal values (Eckerson 2009). In terms of QoE, there are two kinds of indicators, as discussed in Chapter 2, objective KPIs and subjective KPIs (known as KQIs), both of them are considered as measurements tools. In this research, KPIs and KQIs are considered from a quantitative perspective, since the provided dataset is produced with integer and float data types.

As a fundamental part of the performance management cycle, the KPIs are developed through a process that creates, deploys and ultimately evaluates each one to determine and quantify the gap between the actual and target performance indicators. The measure is computed as ratio, expressed as a percentage. This concept is utilised in Kan et al. (2001) as a release-to-release comparison technique for assessing the quality of the software testing process. Also, it has been adopted by Chang et al. (2010) as a paired comparison technique for evaluating and quantifying QoE of online games under various network situations. Consequently, this research utilises Actual-Versus-Target approach as a performance management technique that examines the relation between the KPIs and KQIs, as well as quantifying QoE.

4.3 Iteration 1: Quantification of QoE

4.3.1 Design of Iteration 1

In refereeing to the challenge addressed in the literature regarding the quantification of QoE (e.g. Alreshoodi & Woods 2013; Aroussi & Mellouk 2014), Iteration 1 involves adopting the Actual-Versus-Target approach to quantify the relationship between the objective and subjective factors of QoE. Consequently, a design decision was taken to explore systematically the correlation between the KPIs and KQIs measurements, corresponding to points on a positive coordinate axis, as shown in Figure 4-1, where the x-axis represents the measurement of the objective factors and the y-axis subjective ones. The coordinates of the origin (0, 0) indicate the initial points of the (KPI, KQI) indicators. Each increment on the x and y axes represents the actual measured values of the KPIs and KQIs. The default maximum values on both the x and y axes are considered as target values that are variables and based on business-oriented parameters defined within a Service Level Agreement (SLA) (Batteram et al. 2014), i.e. they can be set to different service providers' requirements and standards. The correlation between the measured values of the KPIs and KQIs forms a square that is expressed by Fa (actual). While, the correlation between the targets forms a square that is expressed by Ft (target). The gap between Fa and Ft can be measured by the Actual-Versus-Target approach (Kan et al. 2001), which has the ability to determine the relative strength and weakness of a particular observation and to make a comparative judgment between what is actually measured (Fa) and what is targeted (Ft). The ratio of Fa and Ft expresses the value of QoE, which can be translated into quantifiable value, as shown in the following formulae:

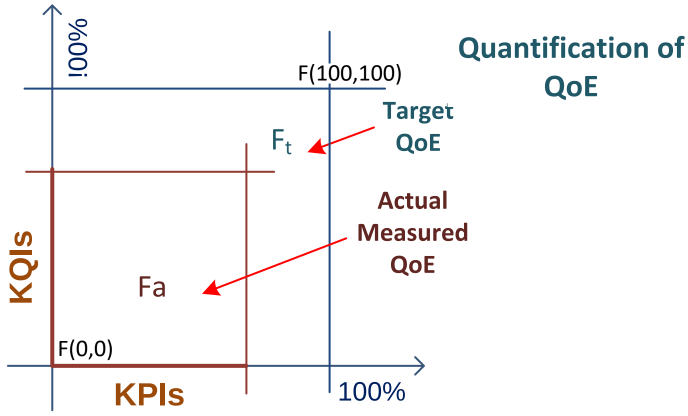


Figure 4-1: Actual-Versus-Target approach for quantifying QoE

$$F_t = F_{t\,(KPI)} * F_{t\,(KQI)} \qquad (1)$$

$$F_a = F_{a\,(KPI)} * F_{a\,(KQI)} \qquad (2)$$

$$QoE = \sum_{k=1}^{n} \frac{F_a}{F_t} \quad \% \qquad (3)$$

The comparison between Fa and Ft ensures the compliance of service quality with user needs. Assuming equality of Fa and Ft, the result indicates that QoE approaches the ideal level. However, a gap between the Fa and Ft value highlights that QoE needs to be improved (Qianqian Xu et al. 2010). Within this design, each factor (e.g. performance, reliability, availability, etc...) can be evaluated separately and the sum provides the overall QoE value. This enables the service provider to determine the factors that may influence QoE and as a result, prioritise their importance (Schumacher et al. 2010).

In term of consistency between the KPI and KQI values, the design decision taken was to systematically evaluate the consistency between them (as conceptualised by Yi et al. (2012) and Martinez (2014)). The effects of this decision are that the consistency level is high when: (1) The correlation between KPI and

KQI is positively strong; and (2) the gap between the measured values of KPI and KQI is close. For example, if the measured KPI set is {20, 40, 60, 80, and 100} and the measured KQI set is {20, 40, 60, 80, and 100}, there is both a strong positive correlation between KPI and KQI and a high level of consistency.

The following steps (illustrated in Figure 4-2) are defined to facilitate the measurement and quantification process.

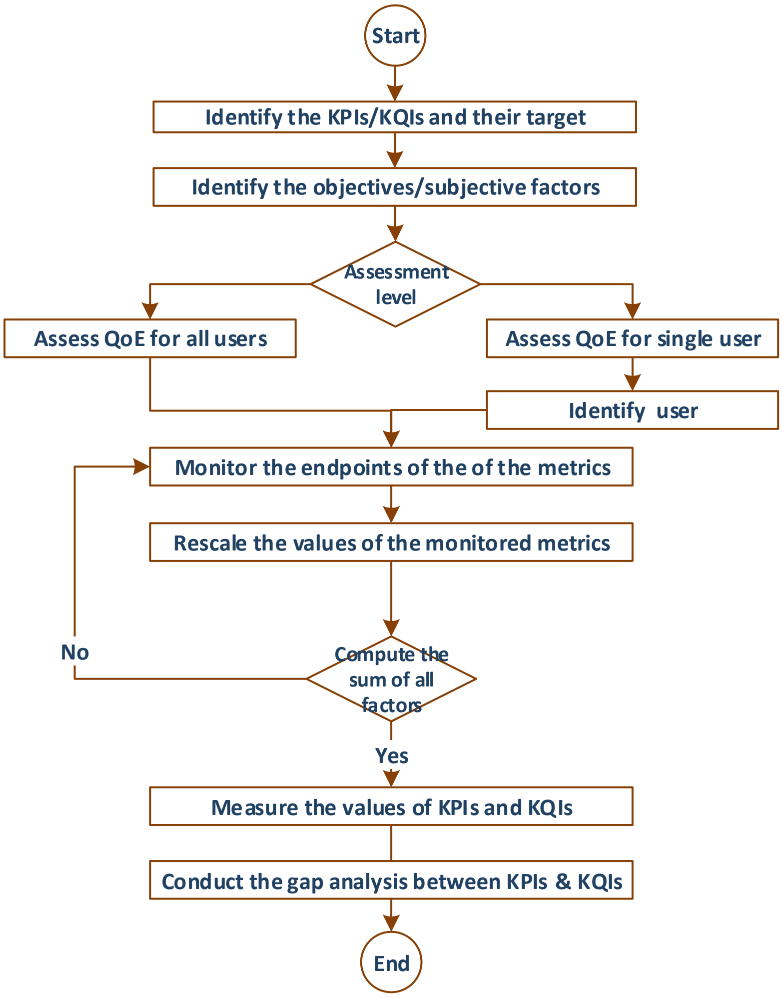


Figure 4-2: Steps for measuring and quantifying QoE

- **Step 1:** Identify the KPIs and KQIs and their targets. The target values are variables based on business-oriented parameters defined within a SLA (Batteram et al. 2014),

- **Step 2:** Identify the objective and subjective factors of the KPIs and KQIs, and set their weights. Each factors is weighted, according to its importance set in the SLA (Behkamal et al. 2009).

- **Step 3:** Monitor the endpoints of each objective and subjective metric. The former is captured from the web services monitoring tools, while the latter is extracted from user profiles (e.g. satisfaction assessment).

- **Step 4:** Rescale the values of the monitored factors and their metrics. To keep consistency between the measurements, the scale of each factor is expressed as a percentage of the ratio of the difference between the measured value and the target value (Yetgin & Göçer 2015).

- **Step 5:** Compute the sum of factors, and measure the values of KPIs and KQIs using the following formulas:

$$\text{KPIs} = \sum F_{obj} \ * \text{Weight} \qquad (1)$$

$$\text{KQIs} = \sum F_{sub} \ * \text{Weight} \qquad (2)$$

- Step 6: Correlate between the KPIs and KQIs. Using the following formula:

$$\text{QoE} = \{\text{KPIs, KQIs}\} \qquad (3)$$

- **Step 7:** Conduct the gap analysis between KPIs and KQIs. The gap between the actual measured and the target values highlights that QoE needs to be improved (Qianqian Xu et al. 2010).

4.3.2 Instantiation of Iteration 1

The implementation is performed with an agile Extreme Programming (XP) development process, which has the capability of breaking the development’s activities into set-by-step increments with minimal advance planning. This is aligned with the DSR methodology as a practical approach (Vidgen et al. 2011) that combines the DSR process with the agile process to develop an efficient software system (Aaen 2008). Following the discussion in Chapter 2, the design in this iteration is implemented by a MVC-based web application combined with 3-tier architecture that separates the web application layers from each other (Tupamäki & Mikkonen 2013). MVC is discussed in more detail in Appendix A. Building upon this, a three layer architecture is developed from the bottom-up as follows: Data Layer, Business Logic Layer, and Presentation Layer.

- **Data Layer**

It manages the data extracted from the middleware server and CRM system through a relational database that contains tables and views created in an Oracle database. The technical data extracted from the middleware server is processed and stored in QOE_OBJ_MONITORING_TB, and the user data extracted from the CRM system is processed and stored in QOE_SUB_MONITORING_TB. The data related to the KPIs and KQIs is retrieved by QOE_KPI_VW and QOE_KQI_VW, whilst the data related to the mapping is retrieved by QOE_KPI_KQI_VW. The Data Definition Language (DDL) of the tables and views is generated as follows.

```
--------------------------------------------------------
--  DDL for Table QOE_OBJ_MONITORING_TB
--------------------------------------------------------
  CREATE TABLE "QOE_OBJ_MONITORING_TB" ("USER_ID" VARCHAR2(30 BYTE), "SESSION_ID"
VARCHAR2(20 BYTE), "FACTOR_CODE" VARCHAR2(5 BYTE), "FACTOR_VALUE" NUMBER(3,0),
"FACTOR_WEIGHT" FLOAT(2))
--------------------------------------------------------
--  DDL for Table QOE_SUB_MONITORING_TB
--------------------------------------------------------
  CREATE TABLE "QOE_SUB_MONITORING_TB" ("USER_ID" VARCHAR2(30 BYTE), "SESSION_ID"
VARCHAR2(20 BYTE), "FACTOR_CODE" VARCHAR2(5 BYTE), "FACTOR_VALUE" NUMBER(3,0),
"FACTOR_WEIGHT" FLOAT(2))
--------------------------------------------------------
--  DDL for View QOE_KPI_VW
--------------------------------------------------------
  CREATE OR REPLACE VIEW "QOE_KPI_VW" ("USER_ID", "KPI_VALUE") AS SELECT
OBJ.USER_ID                              USER_ID,
AVG(OBJ.FACTOR_VALUE * OBJ.FACTOR_WEIGHT)   KPI_VALUE FROM
QOE_OBJ_MONITORING_TB OBJ GROUP BY OBJ.USER_ID
--------------------------------------------------------
--  DDL for View QOE_KQI_VW
--------------------------------------------------------
  CREATE OR REPLACE VIEW "QOE_KQI_VW" ("USER_ID", "KQI_VALUE") AS SELECT
SUB.USER_ID                              USER_ID,
```

```
AVG(SUB.FACTOR_VALUE * SUB.FACTOR_WEIGHT)   KQI_VALUE FROM
QOE_SUB_MONITORING_TB SUB GROUP BY SUB.USER_ID
--------------------------------------------------------
--  DDL for View QOE_KPI_KQI_VW
--------------------------------------------------------
  CREATE OR REPLACE VIEW "QOE_KPI_KQI_VW" ("USER_ID", "KPI_VALUE", "KQI_VALUE") AS
SELECT
KPI.USER_ID                    USER_ID,
KPI.KPI_VALUE                  KPI_VALUE,
KQI.KQI_VALUE                  KQI_VALUE
FROM QoE_KPI_VW KPI, QoE_KQI_VW KQI WHERE  KPI.USER_ID = KQI.USER_ID
```

- **Business Logic Layer**

It manages the core business rules of the QoEWA application, which is implemented in Oracle PL/SQL and Java in accordance with DSR principals (e.g. Hevner & Gregor 2013; Pfeffers et al. 2008). This subsection provides a brief logical presentation via a simplified UML Class model as shown in Figure 4-3 and a UML Sequence model as illustrated in Figure 4-4.

The Class model provides an abstract class called QoE that has an aggregation relationship with the Factor, Metric, and Data classes. The data of monitored metrics are collected via the Data class. Furthermore, there are two core sub-classes reflecting the QoEWA design: (1) The objFactor class that measures the objective factors via the KPI class; and (2) the subFactor class that measures the subjective factors via the KQI class. The result is mapped by the Mapping class, which correlates KPIs and KQIs to quantify QoE via the Quantifying class to predict QoE via the Predicting class and to optimise QoE via Optimising class.

With regards to the interaction, a Sequence model is provided to illustrate the scenario where the QoE assessor intends to assess QoE. The process is initiated at LineLife1. Thereafter, LifeLine2 assesses the objective factors and their KPIs and LifeLine3 does so for the subjective factors and their KQIs. The results are mapped at LineLife10, which invokes the quantifying, predicting and optimisation processes. Each process returns a computed value to the assessor for assessing the overall QoE.

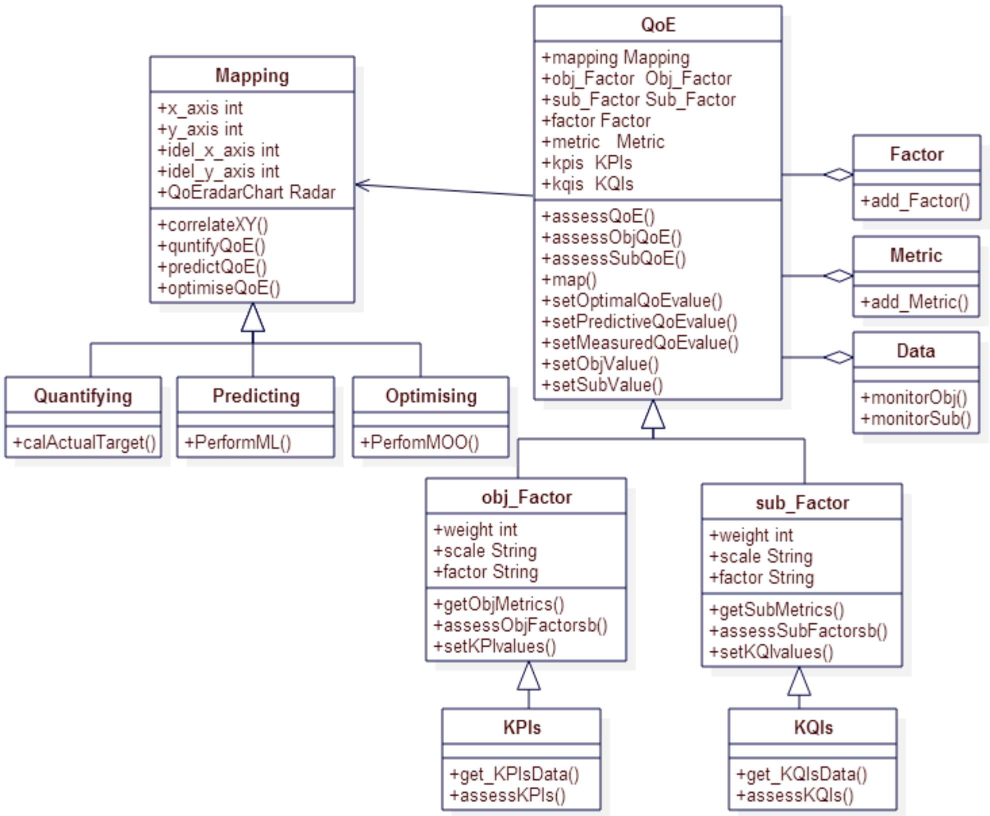


Figure 4-3: UML Class model for QoEWA instantiation

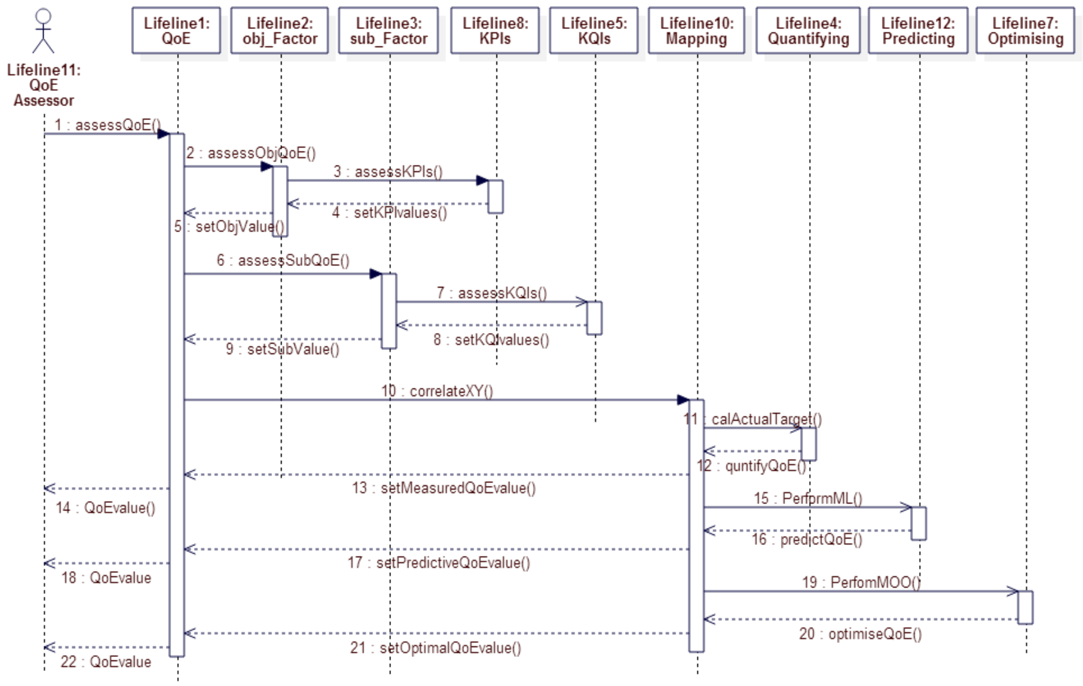


Figure 4-4: UML Sequence model for illustrating the assessment scenario

- **Presentation layer**

It provides a graphical user interface (GUI) that manages the interface of the assessment function provided by the QoEWA model. This is implemented using an Oracle Application Development Framework (Oracle ADF). Figure 4-5 shows a web-based interface with two sections that are provided to the assessor to assess QoE on the application and user level. The logic behind the interface is based on the objects described in the business logic.

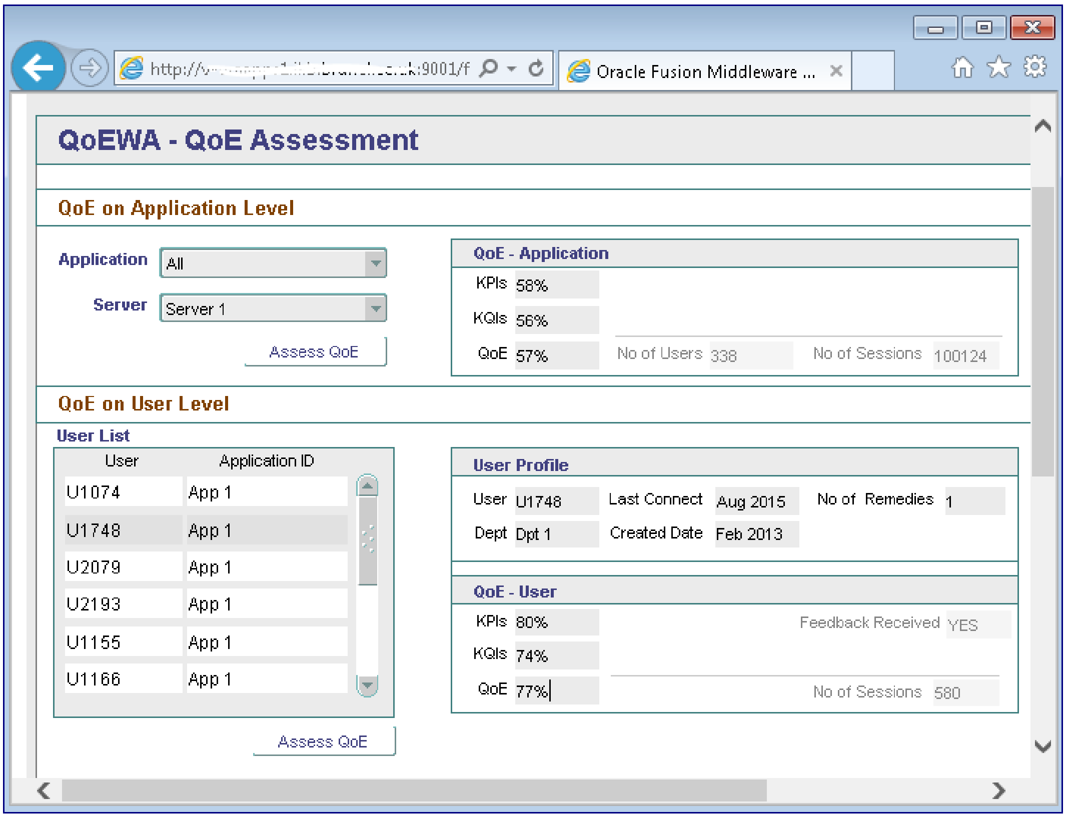


Figure 4-5: The proposed QoEWA interface for quantifying QoE

- **Deployment of QoEWA**

The developed QoEWA was deployed in the context discussed drawing from both a web services and user profile dataset (anonymised data pulled from back-end systems – Oracle Fusion Middleware and CRM system). User sessions are thus tracked during the interaction with the web services and users are polled for their opinion about the quality of the services, i.e. the users have the option to express their opinion online at any time using a web-based poll tool that is integrated with the applications. The poll is designed with questions extracted from factors defined in the conceptual model in Chapter 3. The information of the sessions and polled users is extracted and stored into the Oracle tables, which have been described earlier in the data layer section. Due to the limitation of accessing and extracting some required information from the monitoring tool provided by the web service provider, however, an in-house software tool was developed and installed in the back-end to monitor the session logs, which contain information for diagnosing problems (i.e. the tool has triggers/events created in the database and applications servers to monitor sessions, error, request messages and response time). The principle behind the developed tool comes from the research performed by Al-moayed & Hollunder (2011), which aims to manage the quality of service attributes of Web services. Ultimately, the collected data was combined into a single dataset that is used as input for QoEWA, as shown in the next section.

4.3.3 Application and Testing of Iteration 1

Building upon the inputs provided in Section 3.5.2, the instantiation of the QoEWA was tested in two ways. First, to provide a benchmark in relation to the state-of-art, the correlation between the overall score of KPIs and KQIs was examined based on the assumption that a strong positive correlation indicates an excellent relationship between the objective and subjective factors (Upadhyaya et al. 2014) - examining the relationship between service quality and user satisfaction. Second, the data was run in the context of the full QoEWA model, quantify QoE by comparing the actual values against the target values. A small gap between KPI and KQI values indicates consistency between KPI and KQI, which means that the feedback obtained by user is consistent with (technical) quality of the service.

For the first test, Table 4-1 summarises the R squared value of each factor (F1-F11). The results show that there is a strong positive correlation between the objective and subjective factors, ranging from between R^2= 88 and R^2= 97.

Figure 4-6 shows the overall correlation of these factors, named, the KPIs and KQIs. The overall result shows a high strong correlation of R^2= 96.00

Table 4-1: Summary of the R squared values for each factor

Measurement		F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
KPIs	Mean	60.48	58.94	58.68	59.67	57.63	56.65	56.17	53.80	54.95	57.15	55.11
KQIs	Avg.	61	59.19	55.98	56.93	52.09	56.89	56.41	51.32	52.42	57.39	52.57
	R2	90	96	97	88	96	95	94	97	97	96	93

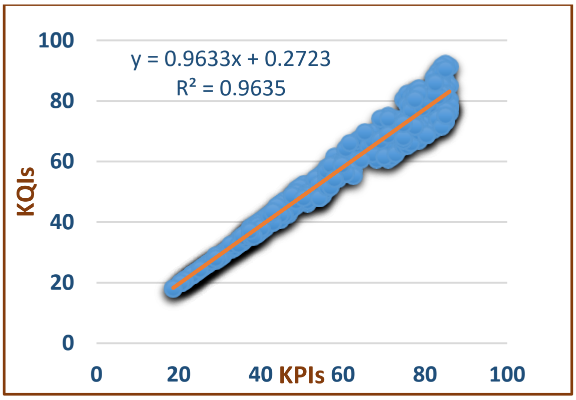


Figure 4-6: Correlation between the measurements of the KPIs and KQIs

Regarding the second test, Table 4-2 summarises the actual-target values of each factor (F1-F11). The results show that there is small gap (2%) between the KPIs and KQIs, thus, according to the assumption above, this indicates a high level of equality and consistency between them. Figure 4-7 shows the actual and target values. The quantitative value of QoE is computed as (52.00 $*$ 51.00) / (86.00 $*$ 92.10) and hence, QoE is 34%. The target value of each factor is strategically driven (Eckerson 2009) by the maximum value that it can reach with respect to the available resources, i.e. in this test, the maximum level of performance can be achieved when the ratio of the completed requests and the unit time is 90.86%.

Table 4-2: Summary of the actual and target values

Value	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
Target KPI	90.86	88.56	88.17	89.66	86.59	85.12	84.39	80.83	82.56	85.87	82.80
Actual KPI	53.83	53.05	52.23	54.90	52.44	50.99	51.68	49.50	50.55	52.01	50.70
Target KQI	99.00	98.19	92.86	94.43	86.40	94.37	93.57	85.14	86.95	95.20	87.21
Actual KQI	54.29	53.27	49.82	52.38	47.40	51.20	51.90	47.21	48.23	52.22	48.36
QoE Value	32%	32%	32%	34%	33%	32%	34%	34%	34%	33%	34%

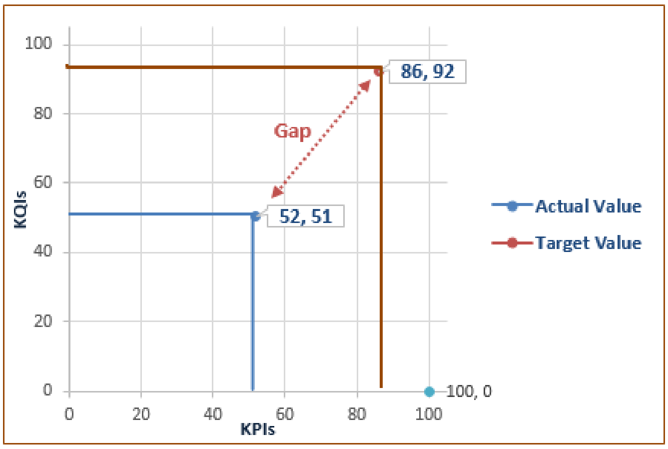


Figure 4-7: Gap analysis based on the Actual-Versus-Target approach

Eventually, after performing both tests, the results are validated by splitting the dataset into four sets, each of which consists of a specific web application that is used by particular users, i.e. four web applications (App 1, App 2, App 3, and App 4). Each set is tested separately in order to examine the correlation between the KPIs and KQIs. In the four tests, a strong positive correlation is found between the KPIs and KQIs as well as the QoE value, which is measured by the Actual-Versus-Target method and is almost constant between 34% and 40%. Furthermore, QoEWA was utilised in terms of improving the QoE of App 1. As a consequence, after tuning the server configuration parameters, e.g. Tune Pool Size parameters, it was found that the improvement of factors related to the KPIs leads to greater user satisfaction, as well as a reduction in the number of remedy requests.

4.3.4 Evaluation of Iteration 1

QoEWA facilitates the correlation between the so called ‘objective’ and ‘subjective’ measurements, which are determined by the KPIs and KQIs. QoEWA develops key components that allow QoE assessors to define QoE factors in accordance with the Web Application architecture. This eventually leads to QoE assessors determining the quality of Web Application components from the service provider and consumer perspectives. The Actual-Versus-Target area obtained from the correlation enables a more holistic measurement of QoE and bridges the gap between the actual measurements and those defined by the Service Level Agreement (SLA). The ratio between the actual and target measurements allows the QoE assessor to monitor QoE easily by dealing with quantified values.

Reflecting this solution back into the problem space, the subjective measurement based on the MOS process (De Koning et al. 2007) (discussed in Chapter 2) for determining the KQIs remains time-consuming and expensive. This is because MOS is conducted by polling users on their satisfaction, without considering factors that may have a strong influence on QoE, such as context, previous experience and scalability (Mirkovic et al. 2014). For example, in this study, it was found that around 60% of all the users did not provide feedback, especially those who worked in a busy and customer-facing environment. Furthermore, it was observed that the majority of provided feedback once did not do so again, which makes tracking user satisfaction impracticable. This issue was actually addressed by Mitra et al. (2011), who developed a dynamic model that evaluates user experience from a QoE perspective in a sequential manner.

Consequently, researchers have started to consider models that enable MOS to be intelligently classified and predicted (Khan et al. 2012; Menkovski, Exarchakos & Liotta 2010; Menkovski, Liotta, et al. 2009; Balachandran et al. 2013). According to the results of a survey conducted by Aroussi & Mellouk (2014), most QoE prediction models are based on the Machine Learning (ML) approach and use an inductive supervised learning approach, where the predictive rules are generated from particular observations or learning. The lesson learned from this evaluation has led the extension of the QoEWA model so as to classify and predict the subjective measurements intelligently. This aspect is addressed by developing another iteration (i.e. Iteration 2) that utilises the Machine Learning (ML) approach in the context of the QoE of Web Application.

4.4 Summary

This chapter has presented an overview of the Actual-Versus-Target approach employed in this research to quantify the relationship between the objective and subjective factors (determined by the KPIs and KQIs). A design decision was taken to: (1) Explore systematically the correlation between the KPI and KQI measurements; and (2) to quantify QoE by making a comparison judgment between what is actually measured and what is targeted. The design has then been implemented following an agile development process involving breaking the activities into set-by-step increments with minimal advanced planning. The developed QoEWA was subsequently deployed and tested twice in two scenarios: (1) To provide a benchmark in relation to the state-of-art by examining the correlation between the KPIs and KQIs; (2) to run the data in the context of the full QoEWA model and to quantify QoE by measuring the actual values versus the target ones. The results show that there is a high strong correlation between the KPIs and KQIs. In addition, there is small gap between them that indicates a high level of equality and consistency between the KPIs and KQIs. In sum, QoEWA develops key components that allow QoE assessors to define systematically QoE factors in accordance with Web Application architecture. Also, The Actual-Versus-Target area obtained from the correlation enables a more holistic measurement of QoE and bridges the gap between the actual measurements and the measurements defined by the Service Level Agreement (SLA). Due to the limitation raised regarding MOS, the lesson learned from this Iteration has led to extension of the QoEWA model to classify and predict the subjective measurements intelligently.

Chapter 5: The Prediction of QoE

5.1 Overview

Based on the evaluation conducted in the initial iteration (Iteration 1 in Chapter 4), this chapter extends the design of QoEWA to enable the prediction of QoE. This chapter is structured as follows: Section 5.2 presents an overview of the Machine Learning approach that is adopted in this research to predict QoE. Section 5.3 presents the development of Iteration 2 through four subsections that describe the design, instantiation, application and tests as well as the evaluation. Section 5.4 presents a summary of this chapter.

5.2 Machine Learning Approach for Predicting QoE

Machine Learning (ML) is a widely used computational approach, based on statistical data analysis, for optimising and improving performance using training data or past experience. It is defined as "a field of study that gives computers the ability to learn without being explicitly programmed" (Samuel 1959 pp. 215). ML is also expressed in an operational terms as follows, “things learn when they change their behaviour in a way that makes them perform better in the future” (Witten et al. 2011). Approaches to ML are either supervised or unsupervised. Supervised learning is based on data collected from past experience to solve regression and classification problems – at its most basic there is an input, X, an output Y and an algorithm to learn the relation(s) between X and Y. Unsupervised learning aims to solve clustering problems – again, at its most basic, there is an input X, an unknown output Y and an algorithm to find the hidden structure of Y by classifying it into meaningful categories (Nilsson 1997). Since, the data is classified and labelled in

Iteration 1, i.e. a supervised learning approach is adopted in this research. Based on the literature review conducted in Chapter 2, it can be observed that there are five supervised learning algorithms that can be adopted for QoE prediction (Mushtaq et al. 2012; Alreshoodi & Woods 2013).

- **Decision Tree (DT):** Is an algorithm generally used to construct a model that classifies the value of a target parameter from several input parameters. The structure of DT has three main elements: (1) The internal node corresponds to the input parameters and tests their values; (2) the branches correspond to the parameter values; and (3) the leaf nodes represent the classification of the target parameter. DT generates rules from each branch, starting from the root node to the leaf node. The rules are expressed as set of nested if-else statements that can be interpreted by programming language (Nilsson 1997; Witten et al. 2011).

- **Naive Bayes (NB):** Is a probabilistic classifier that determines the conditional probabilities of the target dependent variables from the training dataset. NB is based on a set of supervised learning methods that calculate probability by P (c|x) from P(c), P(x), and P(x|c), where P(c|x) is the posterior probability of the dependent target value, P(c) is the prior probability of the target, P(x) is the prior probability of the predictor and P(x|c) is the likelihood, which calculates the probability of the predictor of the target value. NB generates rules of conditional probabilities used for fast training (Nilsson 1997; Witten et al. 2011).

- **K-Nearest Neighbours (KNN):** Is also called instance-based learning with parameter K (IBK) (Witten et al. 2011). KNN is based on the calculation of the distance between instances using the kernel density estimator, i.e. KNN determines the location of the available instances k and then uses a similarity measure for classifying new instance, where k is the nearest neighbour and close to 1 for robust models. The generated result obtained by KNN depends on whether it is used for a classification or regression problem. In classification, the target is assigned to the class among its neighbours, while for regression, the target is assigned to the average k of the similar neighbours (Witten et al. 2011).

- **Sequential Minimal Optimisation (SMO):** Is an algorithm for support vector machine (SVM) learning and quadratic programming problems. SMO breaks the problem down into a set of sub-problems that are solved analytically. It finds a Lagrange multiplier α_1, which is used for the optimisation problem, then picks the next multiplier α_2 to optimise the pair of α_1 and α_2. This process is repeated until convergence is achieved and ultimately updates the SVM, reflecting the new optimal values, i.e. when SMO adjusts the weight of α_1, it must adjust the weight of α_2 (Menkovski, Oredope et al. 2009; Witten et al. 2011).

- **Random Forest (RF):** Is an ensemble algorithm that creates randomised decision trees using a bagging technique to achieve high classification accuracy. It builds a multiple decision tree from a random subset, in which each tree is learned independently on a set of bootstrapped samples. The prediction depends on whether RF is applied for classification or regression. In classification, RF gives a vote for each class and ultimately choose the classification with the highest votes, while in regression, it calculates the mean prediction of the individual trees (Aroussi & Mellouk 2014; Witten et al. 2011).

Most of the studies have consistently shown that DT and RF algorithms provide a high predictive accuracy in classifying the unknowing values of MOS, i.e. the results have shown that these algorithms are most appropriate for predicting the five classes of MOS scores. In contrast, NB and IBK algorithms tend not to perform so well in term of prediction accuracy and classification error rate (Mushtaq et al. 2012; Alreshoodi & Woods 2013; Aroussi & Mellouk 2014). Generally, five statistical tests are used to compare the performance of the classification algorithms, including: True Positive (TP), False Positive (FP), Precision, Recall, and the F-measure (Witten et al. 2011).

- **True Positive (TP):** Refers to the correct classification (sensitivity), and occurs when a statistical test rejects a true hypothesis. The best value of TP is 1.

- **False Positive (FP):** Refers to the incorrect classification (1- specificity) and occurs when a statistical test rejects a true null hypothesis. The best value of FP is 0.

- **Precision:** Measures the accuracy of the classification and relies on the calculation of Precision =TP/ (TP+FP).

- **Recall:** Measures the performance of the classification and relies on the calculation of Recall = TP/ (TP+FN), where FN refers to the false negative value.

- **F-measure:** Combines the recall and precision measurements into a single value to evaluate the accuracy. The value of the F-measure lies between 0 and 1, where 0 indicates the worst classification, while 1 is the best.

Alongside the above testes, a Receiver Operating Characteristic model (ROC) is widely used as a graphical technique for evaluating the performance of the algorithms (Nilsson 1997). The ROC curve illustrates the trade-off between TP and FP (i.e. between sensitivity and specificity). The values of both TP and FP are plotted on ROC space with two-dimensional vectors (x, y), where x is 1-TP and y is FP. The ROC curve starts at (0, 0) and ends at (1, 1). ROC space is separated into two areas of good and poor performance levels, i.e. the area under (0, 1) indicates a good performance level and that under (1, 0) indicates a poor performance level. The ROC curve is used to compare algorithms only if the ROC points are generated from the same dataset, otherwise the scores of the Area Under the Curve (AUC) are convenient for comparing multiple algorithms (Nilsson 1997; Witten et al. 2011).

In addition to the performance and accuracy tests, other aspects need to be considered when preparing and transforming raw data into the training dataset (Witten et al. 2011). For instance, removing noise, missing data, time to train the model, especially for large datasets, large number of attributes and instant prediction all need to be considered. Hence, it is important to understand the context of the problem in terms of dataset structure, its attributes, and what amounts to a reasonable time regarding the prediction process (Kotsiantis 2007; Nilsson 1997). In QoE, data are usually extracted from two data sources, as discussed in Chapter 2, i.e. the network and customer profile and feedback. The size of network data is usually huge, especially in telecoms and internet applications, whereas the size of feedback data is smaller as there are plenty of customers who do not respond. Consequently, there is no single algorithm that can be generalised for QoE prediction problems; each dataset has its own structure that exhibits a particular behaviour (Kotsiantis 2007; Aroussi & Mellouk 2014).

5.3 Iteration 2: Prediction of QoE

5.3.1 Design of Iteration 2

Building upon the literature discussed in Chapter 2 regarding the prediction of QoE (Menkovski, Liotta et al. 2009; Menkovski, Exarchakos & Liotta 2010; Aroussi & Mellouk 2014), a design decision is taken in this iteration to enable the learning and prediction. The design of QoEWA is extended with a function that incorporates a Machine Learning approach, employing five supervised learning algorithms for comparison: DT, NB, SMO, IBK, and RF. The design provides the model with a classifier for predicting the values of the subjective metrics (Mushtaq et al. 2012), which employs the commonly used five-point MOS scale (Excellent, Good, Fair, Poor, and Bad). The classifier is trained with subjective data obtained from the outputs of Iteration 1, where data are statistically analysed, classified and stored in a Customer Relations Management (CRM) system. The theory behind the design is based on the top-down and bottom-up approaches used by Alreshoodi & Woods (2013) for predicting QoE. The top-down approach is based on subjective data collected from the user-side related to the KQIs. Whilst the bottom-up approach is based on objective data collected from the server-side and related to the

KPIs. Both approaches can be applied alongside each other in complementary ways, depending on the data availability and the degree of association between the QoE parameters. For example, the known KPIs enable the model to predict and estimate the unknown KQIs. Figure 5-1 illustrates the correlation between the top-down and bottom-up approaches, where x-axis represents the known data, which belongs to the KPIs, and the y-axis represents the unknown data, which belongs to the KQIs.

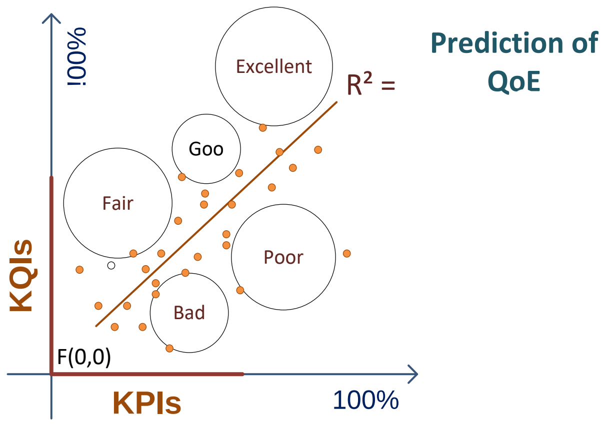


Figure 5-1: Machine Learning (ML) approach for predicting QoE

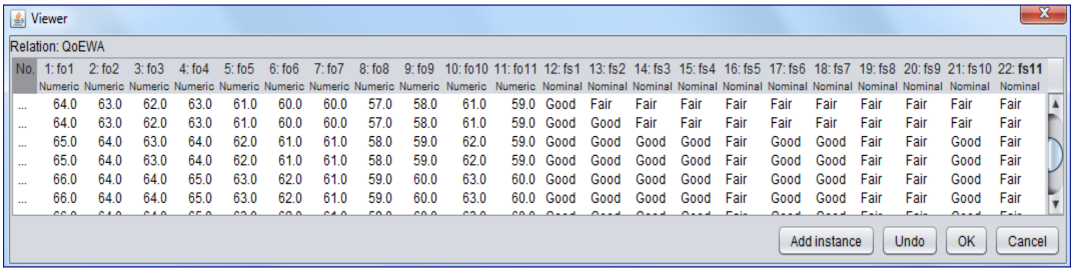
The training dataset includes KPIs and KQIs data, in which the KPIs are expressed as independent variables (feature), whereas the KQIs are labelled dependent variable (target) (Witten et al. 2011). Based on the data extracted in Iteration 1, there are 11 dependent variables corresponded with 11 independent ones. Each independent variable is incorporated into the whole set of the dependent variables to be used as an input of the utilised algorithms. The output of each algorithm is generated as rules that predict the target values, which indicate the values of the MOS scores.

5.3.2 Instantiation of Iteration 2

In implementation terms, the Waikato Environment for Knowledge Analysis (WEKA) tool is utilised in this iteration to implement the chosen ML classifiers via WEKA explorer and WEKA knowledge workflow. WEKA is widely used, not only for research and teaching, but also for commercial applications, because it is an open source tool written in Java under a GNU General Public License and easy to extend and maintain. WEKA has the capabilities of performing ML and data mining tasks, such as filtering, classification, clustering and ranking (Witten et al. 2011).

- **Dataset Structure**

The training dataset used in Iteration 1 is structured and extracted into an Attribute Relation File Format (ARFF), which is provided by Weka. ARFF has two sections, Header and Data (Witten et al. 2011). In the Header section, the objective factors (fo) and subjective factors (fs) are defined as features and target attributes. The features are expressed by {fo1, fo2, fo3, fo4, fo5, fo6, fo7, fo8, fo9, fo10, fo11} and the target attributes are expressed by {fs1, fs2, fs3, fs4, fs5, fs6, fs7, fs8, fs9, fs10, fs11}. Whereas, the Data section contains the raw data of the training dataset, which is subsequently filtered and transformed by ARFFLoade. In order to import data dynamically, a CSVLoader is used to load them from a CSV file that is generated by the database tables developed in Iteration 1. However, in the enterprise and commercial solution, WEKA can be integrated with the Relational Database Management System (RDBMS) (Witten et al. 2011). Figure 5-2 demonstrates the content of the dataset and how it is structured.


Viewer

Relation: QoEWA

No.	1: fo1 Numeric	2: fo2 Numeric	3: fo3 Numeric	4: fo4 Numeric	5: fo5 Numeric	6: fo6 Numeric	7: fo7 Numeric	8: fo8 Numeric	9: fo9 Numeric	10: fo10 Numeric	11: fo11 Numeric	12: fs1 Nominal	13: fs2 Nominal	14: fs3 Nominal	15: fs4 Nominal	16: fs5 Nominal	17: fs6 Nominal	18: fs7 Nominal	19: fs8 Nominal	20: fs9 Nominal	21: fs10 Nominal	22: fs11 Nominal
...	64.0	63.0	62.0	63.0	61.0	60.0	60.0	57.0	58.0	61.0	59.0	Good	Fair	Fair	Fair	Fair	Fair	Fair	Fair	Fair	Fair	Fair
...	64.0	63.0	62.0	63.0	61.0	60.0	60.0	57.0	58.0	61.0	59.0	Good	Good	Fair	Fair	Fair	Fair	Fair	Fair	Fair	Fair	Fair
...	65.0	64.0	63.0	64.0	62.0	61.0	61.0	58.0	59.0	62.0	59.0	Good	Good	Good	Good	Fair	Good	Good	Fair	Fair	Good	Fair
...	65.0	64.0	63.0	64.0	62.0	61.0	61.0	58.0	59.0	62.0	59.0	Good	Good	Good	Good	Fair	Good	Good	Fair	Fair	Good	Fair
...	66.0	64.0	64.0	65.0	63.0	62.0	61.0	59.0	60.0	63.0	60.0	Good	Good	Good	Good	Fair	Good	Good	Fair	Fair	Good	Fair
...	66.0	64.0	64.0	65.0	63.0	62.0	61.0	59.0	60.0	63.0	60.0	Good	Good	Good	Good	Fair	Good	Good	Fair	Fair	Good	Fair

Add instance Undo OK Cancel

Figure 5-2: Sample of the dataset

- **Training and Knowledge Flow**

The dataset is transformed and processed by the Knowledge Flow interface, which provides the components required to configure the inputs and outputs of the chosen ML algorithms (e.g. DT, NB, SMO, IBK, and RF). Each target attribute is assigned with the features from {fo1, fo2, fo3, fo4, fo5, fo6, fo7, fo8, fo9, fo10, and fo11} to provide a batch class as an input to the algorithms. The output of the algorithms is presented by a text viewer and model performance chart. The output includes rules (i.e. a set of nested if-else statements) that are used to predict the MOS of users who did not provide their feedback. In the enterprise and commercial solutions, the Knowledge Flow can be integrated with Extract, Transform and Load (ETL) tools (e.g. Pentaho ETL) to predict the targets values in real-time. To evaluate the performance and accuracy of the algorithms, a Classifier Performance Evaluator is used and the evaluation results are extracted into a CSV file using CSVsaver. Figure 5-3 illustrates the developed Knowledge Flow.

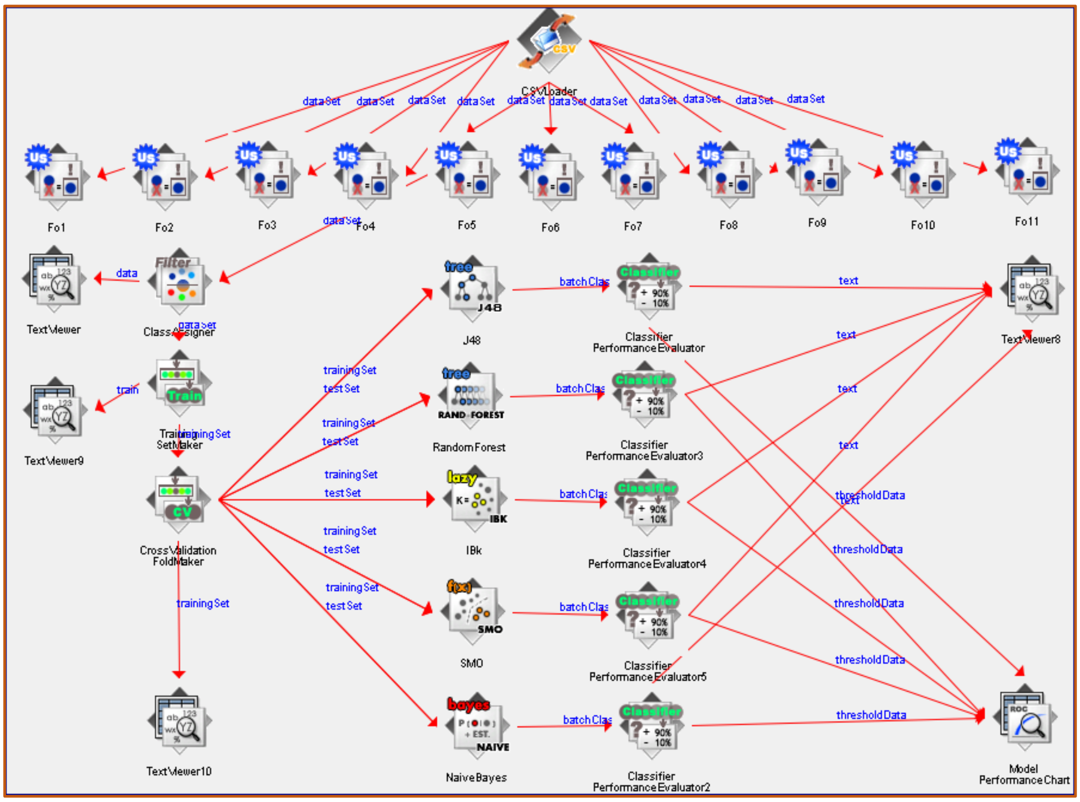


Figure 5-3: Weka Knowledge flow of QoEWA

5.3.3 Application and Testing of Iteration 2

The application of the extended QoEWA is illustrated by a test that examines the scenario where QoEWA is trained on data obtained from previous feedback about the service quality, thus providing a comparative assessment of the ML algorithms for predicting the poll scores. The test examines how well ML algorithms (DT, NB, SMO, IBK, and RF) can perform the QoE prediction process. The test is based on the developed Knowledge Flow which has been described in the previous section. To minimise bias, the experiment performs a 10-fold cross-validation test to evaluate the results of the applied algorithm, which is a generally accepted approach (Mushtaq et al. 2012; Menkovski, Liotta et al. 2009). The Correct Classified Instances (CCI) is used to show the best performance of each ML algorithm and the Mean Absolute Error rate (MAE) is used as a basic indicator to compare the ML algorithms (Menkovski, Liotta et al. 2009; Mushtaq et al. 2012). Table 5-1 shows the percentage of each of the labelled vectors according to the standard MOS scale of each target (factors from Fs1 to Fs11).

Table 5-1: Summary of the labelled and classified vectors

Label	Fs1	Fs2	Fs3	Fs4	Fs5	Fs6	Fs7	Fs8	Fs9	Fs10	Fs11
Excellent	93	81	52	58	23	58	52	17	27	62	27
Good	87	89	108	106	120	106	108	121	116	102	116
Fair	73	80	83	79	90	79	83	95	92	80	92
Poor	64	67	67	67	70	67	67	70	68	69	68
Bad	18	18	25	25	32	25	25	32	32	22	32
Total	*335*	*335*	*335*	*335*	*335*	*335*	*335*	*335*	*335*	*335*	*335*

The developed Knowledge Flow is run for each target and accordingly the results of CCI and MAE are summarised in Table 5-2, to be used as an initial indicator to compare the algorithms. The results are averaged and expressed in Figure 5-4. This shows that the DT algorithm has the minimum absolute error rate (with value 0.07) and in terms of the correctly classified instances, is the best classification algorithm amongst the set employed (with a value of 83.4). Clearly, the difference between the five applied algorithms is small, with a standard deviation of 0.077 for MAE and 0.05% for CCI. The obtained results confirm the conclusion drawn by Mushtaq et al. (2012), who found that DT and RF have higher performance and accuracy than NB, SMO and IBK. Again, as discussed earlier, there is no single algorithm that can be generalised for QoE prediction; each context has its own behavioural model.

Table 5-2: Summary of CCI and MAE of each classifier

	Test	Fs1	Fs2	Fs3	Fs4	Fs5	Fs6	Fs7	Fs8	Fs9	Fs10	Fs11
DT	CCI	0.83	0.79	0.82	0.80	0.90	0.80	0.81	0.88	0.87	0.80	0.87
	MAE	0.07	0.08	0.06	0.11	0.07	0.11	0.08	0.05	0.07	0.07	0.07
IBK	CCI	0.81	0.81	0.79	0.80	0.90	0.80	0.79	0.89	0.88	0.79	0.88
	MAE	0.09	0.09	0.09	0.09	0.06	0.09	0.09	0.06	0.06	0.09	0.06
RF	CCI	0.81	0.81	0.79	0.79	0.89	0.79	0.79	0.88	0.88	0.79	0.88
	MAE	0.09	0.09	0.09	0.09	0.05	0.09	0.09	0.06	0.06	0.09	0.06
SMO	CCI	0.82	0.81	0.80	0.75	0.89	0.75	0.80	0.88	0.87	0.79	0.87
	MAE	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
NB	CCI	0.83	0.81	0.81	0.78	0.89	0.78	0.81	0.88	0.84	0.78	0.84
	MAE	0.07	0.08	0.08	0.08	0.05	0.08	0.08	0.05	0.06	0.08	0.06

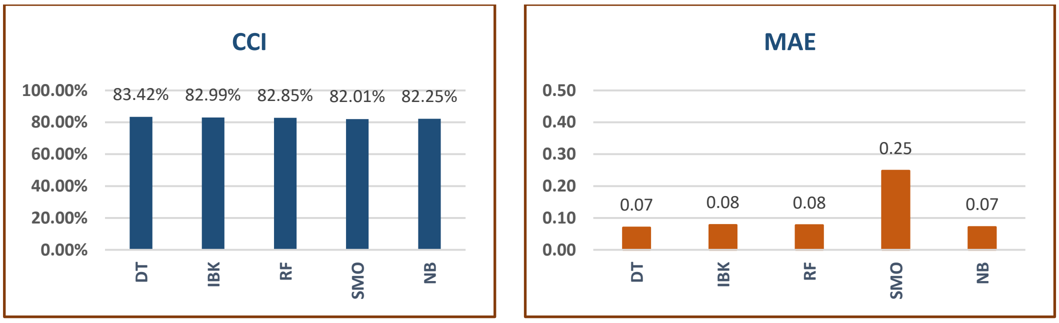


Figure 5-4: Comparison between CCI and MAE

Next, the efficiency of each algorithm is evaluated in Table 5-3. The table summarises the evaluation of the conducted performance and accuracy tests, which were discussed earlier. The results are averaged and expressed in Figure 5-5 for each algorithm. It is found that the DT algorithm gives the best performance accuracy when compared to the others in terms of the TPs, FPs, Precision, Recall, and F-measure as well ROC area, while the SMO algorithm gives the lowest results. However, in general, the evaluation of the five algorithm shows a large ROC area lies between 0.931 and 0.958. This indicates that all the utilised algorithms are able to perform effective prediction on the extracted training dataset and therefore, achieve high sensitivity and low specificity for the prediction. Building upon the results above, DT is considered to be the most efficient algorithm among the others in this research.

Table 5-3: ML test results for each applied algorithm

Classifier	Test	Fs1	Fs2	Fs3	Fs4	Fs5	Fs6	Fs7	Fs8	Fs9	Fs10	Fs11
DT	**TP Rate**	0.83	0.79	0.82	0.80	0.90	0.80	0.81	0.88	0.88	0.80	0.88
	FP Rate	0.06	0.07	0.07	0.07	0.05	0.07	0.07	0.05	0.04	0.07	0.04
	Precision	0.83	0.79	0.82	0.81	0.85	0.81	0.80	0.85	0.87	0.81	0.87
	Recall	0.83	0.79	0.82	0.80	0.90	0.80	0.81	0.88	0.88	0.80	0.88
	F-Measure	0.82	0.79	0.81	0.80	0.87	0.80	0.80	0.87	0.87	0.79	0.87
	ROC Area	0.98	0.98	0.97	0.98	0.97	0.92	0.95	0.97	0.95	0.93	0.95
IBK	**TP Rate**	0.81	0.81	0.79	0.80	0.90	0.80	0.79	0.89	0.88	0.79	0.88
	FP Rate	0.07	0.07	0.08	0.07	0.05	0.07	0.08	0.05	0.05	0.07	0.05
	Precision	0.80	0.80	0.78	0.80	0.85	0.80	0.78	0.85	0.86	0.79	0.86
	Recall	0.81	0.81	0.79	0.80	0.90	0.80	0.79	0.89	0.88	0.79	0.88
	F-Measure	0.80	0.81	0.78	0.80	0.88	0.80	0.78	0.87	0.87	0.79	0.87
	ROC Area	0.95	0.94	0.94	0.94	0.97	0.73	0.94	0.96	0.97	0.94	0.97
RF	**TP Rate**	0.82	0.81	0.79	0.79	0.89	0.79	0.79	0.88	0.88	0.79	0.88
	FP Rate	0.06	0.06	0.07	0.07	0.05	0.07	0.07	0.06	0.05	0.07	0.05
	Precision	0.81	0.81	0.79	0.79	0.85	0.79	0.79	0.85	0.87	0.79	0.87
	Recall	0.82	0.81	0.79	0.79	0.89	0.79	0.79	0.88	0.88	0.79	0.88
	F-Measure	0.81	0.81	0.79	0.79	0.87	0.79	0.79	0.86	0.88	0.79	0.88
	ROC Area	0.96	0.94	0.95	0.95	0.98	0.95	0.95	0.97	0.98	0.95	0.98
SMO	**TP Rate**	0.82	0.81	0.80	0.75	0.89	0.75	0.80	0.88	0.87	0.79	0.87
	FP Rate	0.06	0.06	0.09	0.10	0.05	0.10	0.09	0.05	0.06	0.07	0.06
	Precision	0.84	0.82	0.70	0.64	0.85	0.64	0.70	0.85	0.81	0.78	0.81
	Recall	0.82	0.81	0.80	0.75	0.89	0.75	0.80	0.88	0.87	0.79	0.87
	F-Measure	0.81	0.80	0.74	0.69	0.86	0.69	0.74	0.86	0.84	0.78	0.84
	ROC Area	0.93	0.93	0.92	0.91	0.95	0.91	0.92	0.95	0.95	0.92	0.95
NB	**TP Rate**	0.83	0.81	0.81	0.78	0.89	0.78	0.81	0.88	0.85	0.78	0.85
	FP Rate	0.06	0.06	0.81	0.08	0.05	0.08	0.07	0.05	0.05	0.07	0.05
	Precision	0.84	0.81	0.80	0.77	0.87	0.77	0.80	0.85	0.85	0.78	0.85
	Recall	0.83	0.81	0.81	0.78	0.89	0.78	0.81	0.88	0.85	0.78	0.85
	F-Measure	0.82	0.80	0.80	0.77	0.87	0.77	0.80	0.87	0.85	0.78	0.85
	ROC Area	0.96	0.95	0.95	0.95	0.97	0.95	0.95	0.97	0.97	0.95	0.97

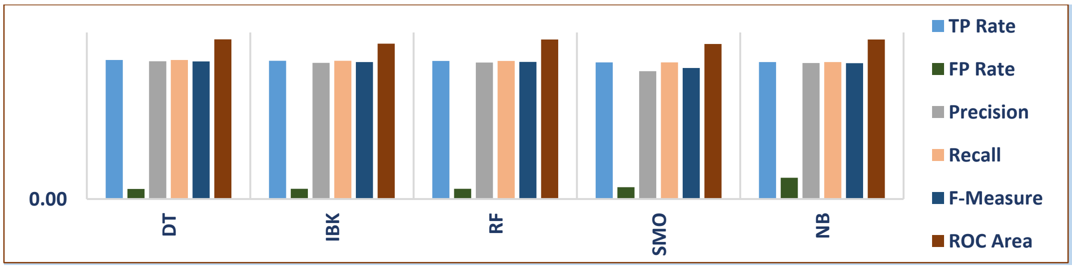


Figure 5-5: Comparison between the ML tests

Based on the evaluation and trained data obtained from the DT algorithm, its output is expressed as rules that can be programmatically developed as nested if-else statements, as shown in Table 5-4. Hence, the control structure of the if-else specifies the inputs, which are based on the KPIs, while the decision structure of the if-else generates the outputs, which are interpreted as MOS for those who do not provide their feedback on the quality of the provided services. The MOS values are then translated to KPI values according to the design decision presented in Chapter 3 and Chapter 4.

Table 5-4 Generated rules of the DT algorithm

<table>
<tr><th>Target</th><th>Rule</th><th>Target</th><th>Rule</th></tr>
<tr><td>Fs1</td><td>fo4 <= 57.5
| fo1 <= 39.51
| | fo1 <= 24.69: Bad (21.0/1.0)
| | fo1 > 24.69: Poor (52.0/1.0)
| fo1 > 39.51: Fair (68.0/8.0)
fo4 > 57.5
| fo1 <= 80: Good (67.0/13.0)
| fo1 > 80: Excellent (94.0/22.0)</td><td>Fs7</td><td>fo4 <= 43.85
| fo1 <= 26.67: Bad (27.0/1.0)
| fo1 > 26.67: Poor (58.0/2.0)
fo4 > 43.85
| fo4 <= 68.22: Fair (86.0/11.0)
| fo4 > 68.22
| | fo4 <= 86.73: Good (106.0/27.0)
| | fo4 > 86.73: Excellent (25.0/9.0)</td></tr>
<tr><td>Fs2</td><td>fo4 <= 57.5
| fo10 <= 39.2
| | fo1 <= 23.7: Bad (20.0)
| | fo1 > 23.7: Poor (60.0/4.0)
| fo10 > 39.2: Fair (61.0/3.0)
fo4 > 57.5
| fo4 <= 77.96: Good (63.0/12.0)
| fo4 > 77.96: Excellent (98.0/33.0)</td><td>Fs8</td><td>fo10 <= 44.8
| fo1 <= 28.64: Bad (37.0/1.0)
| fo1 > 28.64: Poor (64.0/3.0)
fo10 > 44.8
| fo1 <= 72.1: Fair (78.0/5.0)
| fo1 > 72.1: Good (123.0/20.0)</td></tr>
<tr><td>Fs3</td><td>fo4 <= 43.85
| fo1 <= 26.67: Bad (27.0/1.0)
| fo1 > 26.67: Poor (58.0/2.0)
fo4 > 43.85
| fo4 <= 68.22: Fair (86.0/11.0)
| fo4 > 68.22
| | fo4 <= 86.73: Good (106.0/27.0)
| | fo4 > 86.73: Excellent (25.0/9.0)</td><td>Fs9</td><td>fo10 <= 44.8
| fo1 <= 28.64: Bad (36.0/1.0)
| fo1 > 28.64: Poor (62.0/4.0)
fo10 > 44.8
| fo1 <= 74.07: Fair (90.0/11.0)
| fo1 > 74.07
| | fo1 <= 88.89: Good (93.0/7.0)
| | fo1 > 88.89: Excellent (21.0/8.0)</td></tr>
<tr><td>Fs4</td><td>fo4 <= 58.47
| fo10 <= 42
| | fo1 <= 26.67: Bad (26.0)
| | fo1 > 26.67: Poor (59.0/2.0)
| fo10 > 42: Fair (61.0/1.0)
fo4 > 58.47
| fo4 <= 85.76: Good (125.0/39.0)
| fo4 > 85.76: Excellent (31.0/8.0)</td><td>Fs10</td><td>fo10 <= 39.2
| fo1 <= 25.68: Bad (22.0)
| fo1 > 25.68: Poor (58.0)
fo10 > 39.2
| fo4 <= 68.22
| | fo1 <= 59.26: Fair (65.0/1.0)
| | fo1 > 59.26: Good (27.0/12.0)
| fo4 > 68.22
| | fo4 <= 85.76: Good (97.0/33.0)
| | fo4 > 85.76: Excellent (33.0/8.0)</td></tr>
<tr><td>Fs5</td><td>fo10 <= 44.8
| fo1 <= 28.64: Bad (36.0/1.0)
| fo1 > 28.64: Poor (65.0/3.0)
fo10 > 44.8
| fo1 <= 72.1: Fair (80.0/5.0)
| fo1 > 72.1: Good (121.0/17.0)</td><td>Fs11</td><td>fo10 <= 44.8
| fo1 <= 28.64: Bad (36.0/1.0)
| fo1 > 28.64: Poor (62.0/4.0)
fo10 > 44.8
| fo1 <= 74.07: Fair (90.0/11.0)
| fo1 > 74.07
| | fo1 <= 88.89: Good (93.0/7.0)
| | fo1 > 88.89: Excellent (21.0/8.0)</td></tr>
<tr><td>Fs6</td><td>fo4 <= 58.47
| fo10 <= 42
| | fo1 <= 26.67: Bad (26.0)
| | fo1 > 26.67: Poor (59.0/2.0)
| fo10 > 42: Fair (61.0/1.0)
fo4 > 58.47
| fo4 <= 85.76: Good (125.0/39.0)
| fo4 > 85.76: Excellent (31.0/8.0)</td><td></td><td></td></tr>
</table>

5.3.4 Evaluation of Iteration 2

Incorporating an ML approach into QoEWA, enables the model to predict and evaluate subjective data dynamically (via MOS), based on limited user data, in a manner that builds a training dataset intelligently from a few samples. Based on the dataset obtained from Iteration 1, the developed ML generates a set of decision rules to classify data (e.g. Excellent, Good, Fair, Poor, and Bad). Furthermore, ML enables QoEWA to facilitate the relationship between the objective and subjective factors and as a result allowing for the mapping between the KPIs and KQIs. Reflecting this back into the problem space, QoEWA, with the incorporated ML rules, enables the QoE assessor to understand the links and requirements that bridge between the service quality and user experience.

In terms of decision making, ML alone does not provide a prescription for controlling and optimising QoE, i.e. it generates a set of decision rules, without specifying the range values of the factors, which can lead to the optimal QoE. Moreover, it is important for the QoE assessor not only to predict the unknown KQIs and the bridge between these and the KPIs, but it is also to refine and adjust the KPIs in accordance with user requirements and the resources available (Yi et al. 2012; Martinez 2014). The ultimate goal of QoE assessment is to ensure that the users are satisfied and that resources are well controlled and efficiently managed (Elkotob et al. 2010; Baraković & Skorin-Kapov 2013).

Building upon the above, there is a significant need to develop another iteration that extends QoEWA in a manner that enables the model to control and optimise QoE, and provide an acceptable level of user experience with minimal technical resources. This falls in line with the Multi-Objective Optimisation MOO approach, which is widely used for optimisation problems (Ivesic et al. 2011). In terms of QoE, MOO can be utilised to adjust the balance between user experience and network resources (Ivesic et al. 2011; Baraković & Skorin-Kapov 2013). Consequently, the MOO approach is incorporated into QoEWA, as shown in the next chapter, to optimise QoE by adjusting the balance between the KPIs and KQIs with respect to resource available and user experience requirements.

5.4 Summary

This chapter has presented an overview of the ML approach incorporated into QoEWA as a technique for developing Iteration 2. Based on a review of relevant research on QoE prediction, five supervised learning algorithms were compared: DT, NB, SMO, IBK, and FR. The design provides the model with a classifier for predicting the values of the subjective metrics (expressed by the KQIs). The training dataset includes KPIs and KQIs data, in which the KPIs are expressed as independent variables (feature), whereas the KQIs are labelled dependent variables (target). The output of each algorithm is generated as rules that predict the target values which indicate the value of the MOS scores. The design of Iteration 2 is implemented by WEKA tool to examine the chosen ML classifiers. The dataset is transformed by the Knowledge Flow interface, which provides components that process the inputs and outputs of the chosen ML algorithms. The developed Knowledge Flow is run for each target and accordingly the results of CCI and MAE are calculated to determine the efficiency of each algorithm. The results obtained from the run show that the DT algorithm is the most efficient algorithm when compared to the others. However, from the evaluation, it is found that the prediction alone does not provide a solution for optimising QoE, i.e. ML generates a set of decision rules, without definition, which can lead to the optimal QoE value. This leads to extension of the design of QoEWA and the development of Iteration 3 (Chapter 6) for enabling QoEWA to control and optimise QoE.

Chapter 6: The Optimisation of QoE

6.1 Overview

The evaluation of Iteration 2 leads to the extension of the design of QoEWA in this chapter to optimising and controlling QoE. This chapter is structured as follows: Section 6.2 presents an overview of the MOO approach, which is adopted in this research to optimise and control QoE. Section 6.3 presents the development of Iteration 3 through four sub sections that describe the design, instantiation, application and tests and evaluation. Section 6.4 presents a summary of this chapter.

6.2 Multi Objective Optimisation Approach for Optimising QoE

Several models have been proposed in the literature to determine the optimal QoE value that maximises the perceived user experience and minimises the technical resources. However, few have considered the nature of the relation between perceived user experience and technical resources, as well as the optimisation problem, which can be efficiently solved by a variety of approaches, such as optimisation through rate adaption, cross-layer design, scheduling or content and resource management. As shown in Section 2.3.5, each approach focuses on different scope, metrics and adaption methods (Qadir et al. 2015). According to Baraković & Skorin-Kapov (2013), all QoE optimisation approaches have one fundamental aim, which is to evaluate and adjust the balance between different objectives of QoE. In fact, this aim can be formulated within the Multi-Objective Optimisation (MOO) approach, which is widely used for optimisation problems (Ivesic et al. 2011).

MOO is based on mathematical optimisation models that deal with multiple objective functions needing to be optimised in a simultaneous manner (Marler & Arora 2004). It has been commonly applied in engineering, science, economic, and other disciplines as a multiple criteria decision making technique to decide the trade-off between multiple conflicting objectives that have multiple candidate solutions (Marler & Arora 2004; Deb 2011) (e.g. to determine the alternatives when quality is maximised and resources are minimised). MOO is computed in different ways and from different perspectives and angles, such that it can be employed as a mechanism to minimise both objectives or maximise both objectives, or maximise one objective and minimise the other (Fleming et al. 2005). Hence, there is no totalitarian solution that simultaneously optimises objectives, however, according to Marler & Arora (2004) any typical MOO solution has to find a set of points that demonstrate the optimal trade-off between objectives. This is commonly determined by the Pareto optimality technique, where the aim is to find solutions of a vector optimisation problem to help the decision maker to find the acceptable decisions with respect to the economic aspects.

Generally, MOO is constructed to include multiple objectives to be optimised simultaneously, with the association of equality and inequality constraints (Konak et al. 2006). In mathematical terms, the MOO problem is defined as a function F that associates two sets of constraint variables and objective values, which is expressed as follows:

$$Min \quad y = F(x) = [F_1(x), F_2(x), \dots, F_n(x)]$$

$$subject\ to \quad h_i(X) \leq 0, i = 1,2, \dots p; \quad g_i(X) = 0, j = 1,2, \dots q$$

$$x = [x_1, x_2, \dots, x_n] \in R^n$$

$$y = [y_1, y_2, \dots, y_n] \in R^m$$

Where, F(x) is a vector, and x indicates the dimensional vector n, which represents the decision variables. Whereas y indicates the dimensional vector m, which represents the target variables (objective functions). P and q signify the number of equality and inequality constraints. R represents the space of the decision vectors and target vectors (Konak et al. 2006; Marler & Arora 2004; Andersson & Andersson 2000). The best decision vectors are defined as a Pareto optimal solution, which is mathematically expressed as:

$$x^* \in \mathrm{S}, \text{where S is the feasible solution set}$$

$$x^* \text{ is a Pareto optimal solution if and only if } F_{\mathrm{i}}(x) \geq F_{\mathrm{j}}(x^*)$$

$$\text{where j} = 1,2,\dots,\text{n and j} \neq \text{i}$$

Hence, the decision vector is a Pareto optimal solution, if and only if there is no other decision vector exists in the feasible solution set S that improves at least one objective without causing a simultaneous decrease to other objectives. The set of a Pareto optimal solution in the objective space is called a non-dominated Pareto optimal solution set that forms a so called ‘Pareto Front’, which determines the trade-off between objectives by plotting alternative solutions lying at points on the Pareto curve (Marler & Arora 2004).

In real-life optimisation problems, MOO is employed in a variety of contexts (Marler & Arora 2004; Ngatchou et al. 2005), as mentioned above. However, since this chapter is particularly concerned with QoE optimisation and is aimed at allocating resources in an efficient manner with respect to user experience and satisfaction, MOO is employed as a multi-criteria technique for such optimisation (Andersson & Andersson 2000). Whilst a Pareto Efficient Frontier (PEF) is used in the design section as a curve that represents the trade-off between the KPIs and KQIs.

To obtain the PEF, there are several techniques in literature based on a stochastic search approach developed to find the Pareto optimal solutions set, such as numerical and guided random techniques (Andersson & Andersson 2000; Ngatchou et al. 2005). However, the most popular technique used for MOO is the

Genetic Algorithm (GA), which is basically derived from evolutionary theory (Konak et al. 2006). In GA terms, a solution vector is called a chromosome, which is formulated from a discrete unit knows as a gene. A chromosome corresponds to one solution from the solution space. The mapping between a chromosome and the solution space is called encoding and the collection of chromosomes are operated in a population that is normally randomly initialised. GA has two operators that generate new chromosomes from existing ones: Crossover and mutation (Konak et al. 2006). The crossover operator combines two chromosomes, called parents, to generate new chromosomes, called offspring. The idea behind the offspring is that the new chromosome may have better value than the two parents, if the offspring inherits the best characteristics from each parent. The crossover operator is repeated until the pool of parents is empty. The mutation operator applies random changes to the chromosomes' characteristics to expand the chromosome pool (at gene level) and to generate optional chromosomes with improved fitness. Both crossover and mutation are variation modules that determine a set of solutions and then systematically or randomly generate potential better ones (Konak et al. 2006; Deb 2011). In sum, GA has five basic steps including:

- **Initialisation:** To configure the number of chromosomes and rate values of mutation and crossover, thereafter generating the chromosomes with random values in the population;
- **Evaluation:** To evaluate the fitness of the initialised chromosomes and determine the most optimal solutions to be kept in memory (a population);
- **Selection:** To select two parents of chromosomes from memory, the better the fitness is the more appropriate for selection;
- **Crossover:** To crossover the chromosomes to find parents that create new offspring that have better characteristics than the parents;

- **Mutation:** To apply random changes to the segments of the chromosome to expand the chromosome pool and to increase the variety of chromosomes and as a result, improve the solution.

6.3 Iteration 3: Optimising and Controlling QoE

6.3.1 Design of Iteration 3

Based on the discussion above, a design decision was taken in this iteration to control the balance between resources and user experience. Accordingly, the existing design of QoEWA is extended by incorporating the MOO approach into the model so as to examine the trade-off between multiple objectives (KPIs and KQIs). Figure 6-1 illustrates the correlation between the KPIs and KQIs and their correlated values, which are formulated in MOO as decision vectors and the space of these vectors.

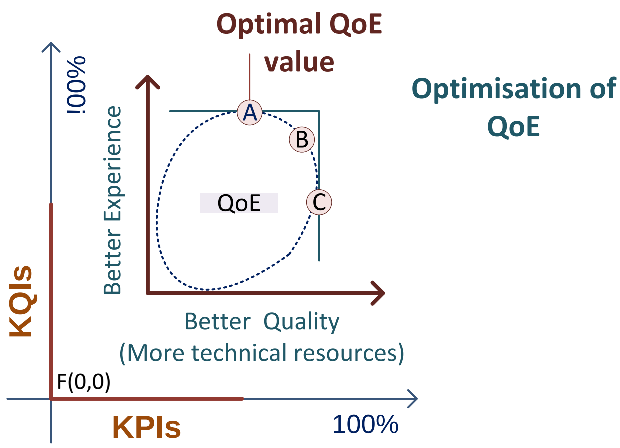


Figure 6-1: Multi-Objective Optimisation problem for optimising QoE

The curve formed by points (A), (B) and (C) generates a PEF (Andersson & Andersson 2000), and consists of two objective KPIs and KQIs, where it is not feasible to make one objective better-off without making the other one worse-off (Sharma et al. 2012). In terms of QoE, point (A) is considered to be the best optimal solution, because it agrees with the notion that QoE is optimised when the service is provided with minimal technical resources at an acceptable level of satisfaction (Agboma & Liotta 2008). While, the alternatives ranging between (B) and (C) are considered as multi-criteria options that can be considered when there are sufficient resources for service improvement. Hence, the design here presents alternative solutions (A, B, and C) to allow the decision-maker interactively to explore and evaluate the alternatives via multi-criteria decision-making.

For determining the non-dominated solutions that correspond to the trade-off between KPIs and KQIs, it was found from the overview presented in the previous section that GA is a well suited search technique for solving multi-criteria optimisation problems (Konak et al. 2006). Consequently, GA is incorporated with the MOO formula to generate decision and target variables based on the steps illustrated in Figure 6-2:

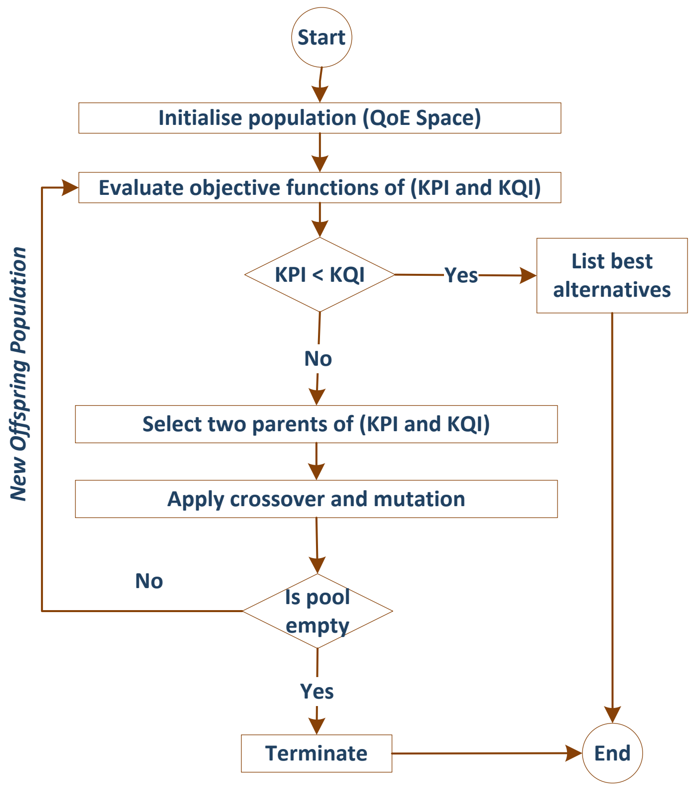


Figure 6-2: Flowchart of the optimisation process

- **Step 1:** Initialise a random population of corresponding values of KPI and KQI. The KPI values are extracted from a real dataset (dataset used in

Iteration 1). While, the KQI values are predicted by the rules obtained from Iteration 2.

- **Step 2:** Evaluate the fitness of the initialised solutions and determine the most optimal solutions to be kept in memory. Thus, since the aim is to minimise KPI and maximise KQI, the memory stores only values where KQI is greater than KPI.

- **Step 3:** Select two corresponding values (parents) of KPI and KQI from the population according to their fitness, the better the fitness is the more appropriate for selection, e.g. if the first value is (50, 60) and the second value is (50, 70), then he second value is more appropriate since the value of KQI is greater than the value of the first one.

- **Step 4:** After selecting the solutions, the position of the crossover point is determined (Konak et al. 2006) by generating random numbers between the minimum and maximum values of the KPI, while the KQI values are predicted based on the defined rules obtained from Iteration 2. Thereafter, the mutation process applies the random values of KPI and KQI into the pool to increase the variety of chromosomes (Konak et al. 2006; Deb 2011).

Ultimately, this design is utilised to build a simulation system that generates a set of points to generate the non-dominated Pareto optimal solutions, which in fact demonstrate the optimal trade-off between the KPIs and KQIs. The details of the implementing of the simulator are described in the next section.

6.3.2 Instantiation of Iteration 3

The instantiation activity is constructed and performed by Netlogo tool, which is presented by Wilensky (1999) as a java-based programmable modelling environment for both research and education. It is available across a wide range of disciplines, because it has been released free under General Public Licence (GPL).

NetLogo has been developed with a large library of sample models and features, including programming and visualisation features.

Moreover, it has the capability of exploring the space of optimisation problems using genetic algorithms. Thus, there are several NetLogo models proposed for examining evolutionary fitness and natural selection problems, such Genetic Drift models (Wilensky 1999). Hence, for implementing Iteration 3, the design presented earlier is translated into a computer-based system based on NetLogo that includes the following key components:

- **GUI Component:** It is a form-based interface that includes a number of objects that control the system functions. Figure 6-3 displays the main user interaction elements, which include: (1) Sliders (F1, F2, F3, F4 …and F11) to initialise and adjust the KPI values of the objective factors. Each slide is associated with the factor's weight and the if-statement formulated by ML to predict the KPI values; (2) Buttons that invoke the main processes which are identified in the design, i.e. setup, exploration and optimisation process; and (3) plotting features to display the outputs.

- **Data Component:** It is based on objective and subjective data. The former are extracted from the realistic dataset used in Iteration 1, while the latter are predicted by the rules formulated via the ML algorithms conducted in Iteration 2. The dataset is uploaded by the CSV-Extension, which allows the ABM system to update dynamically the parameters used to initialise the number of agents as well as the minimum, maximum and average values of KPI. The initialised KPI values can be adjusted by the sliders, in which new values are initialised by the assessor.

- **Output Component:** The output is generated by two processes. First, in the exploration process the output is generated by a plot that presents the correlation between the KPI and KQI values. Second, in the optimisation process the output is generated by a plot and command centre that determines the Pareto curve, including the optimal values A, B, and C. During the optimisation process, the system runs a command that determines the values of the objective factors (f1-F11), where the value of KPI is less than the value of KQI, to produce an alternatives list.

- **Optimisation Component:** Executes the core code, which has two main functions: (1) The correlation function, which determines the correlation between KPIs and KQIs; and (2) the optimisation function, which generates random values based on GA to produce the Pareto curve discussed earlier in the design section. The input component provides the initial values of KPI for the optimisation process, while the output component generates the values produced from the optimisation component through the GUI component.

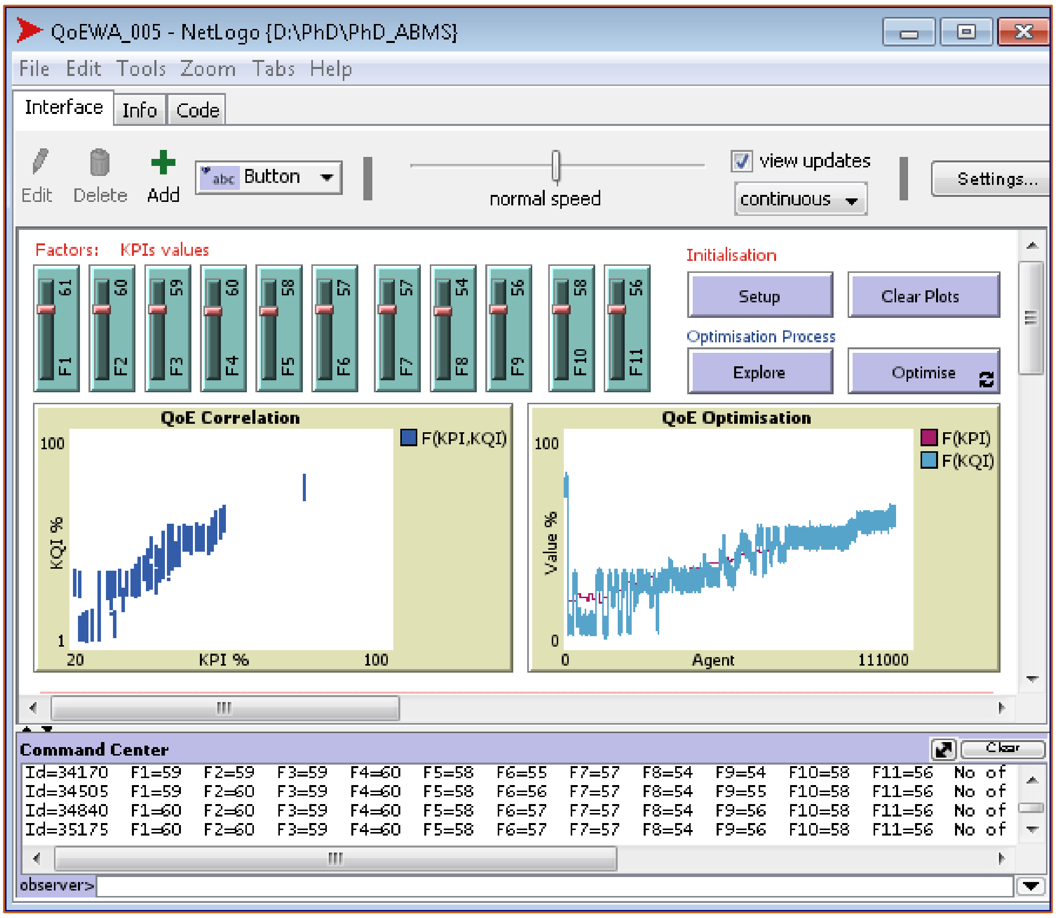


Figure 6-3: Extending the implementation of QoEWA using the NetLogo

6.3.3 Application and Testing of Iteration 3

The instantiation of Iteration 3 is implemented as an extension of QoEWA and applied in the context of Web Application as it is performed in Iteration 1 and Iteration 2. The developed NetLogo simulator imported the 335 records from the dataset produced from Iteration 1. Alongside this data, the simulator imports the predictive rules that were formulated in Iteration 2 by ML to predict the value of KQIs. Accordingly, two tests are performed. First, to validate the model by exploring the data and comparing it with the analysis obtained in Iteration 1. Second, to compute and predict points A, B, and C, i.e. the optimal QoE value.

In the first test, the correlation between the KPIs and KQIs is determined, the output shows a strong positive correlation between their values, with R^2=0.97, as shown in Figure 6-4. This result is consistent with the R^2 value obtained in Iteration 1. Hence, this outcome confirms the validity of the simulator, where the predicted result is fairly consistent with the realistic one.

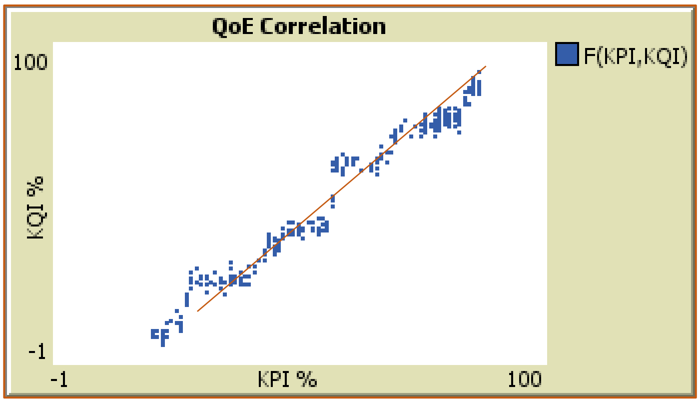


Figure 6-4: QoE correlation

In the second test, the optimal values are calculated based on the design concept suggested earlier. The simulator generates the PEF curve (Points A, B, and C), as shown in Figure 6-5. The result (at Point A) is considered to be the best optimal value, because it agrees with the notion that QoE is optimised when the service is provided with minimal technical resources at an acceptable level of satisfaction (Agboma & Liotta 2008). It is found that the best optimal value of KQI is 96%, which is obtained when the KPI value exceeds 85%.

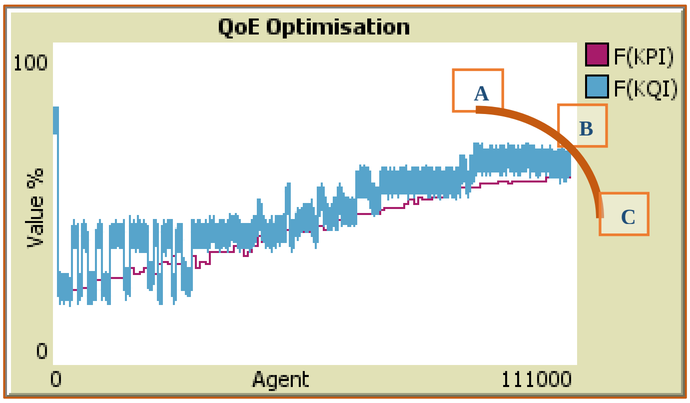


Figure 6-5: MOO optimisation for QoEWA

6.3.4 Evaluation of Iteration 3

Based on the design decision taken in this iteration, the Pareto set is generated by a simulator to determine three points (A, B, and C), where A is taken as the best alternative value in terms of user experience and resource used. At point A, resources deployed are significantly less than at point B and C. The balance between user experience and resource is indicated by the balance between the KPIs and KQIs, which depends on user and business requirements. That is, if the business goal is to improve quality and reduce resource, the alternative at point A is the best. However, if the goal is to improve both the experience and quality of service, the alternative at point C is the best. To decide which alternative is more appropriate to be chosen (A, B, or C), the KPIs and KQIs need to be assigned to user and business requirements, i.e. the actual measured values need to be compared with the expected values.

Since this work was part of a solution that practically assesses QoE, the results obtained from the simulator were analysed and formulated so as to identify the factors behind lower satisfaction ratings. Based on the tests conducted in this section, it has been found that factors (F1, F2, F10, and F11) are important for gaining better performance of QoE. This therefore led the solution architect (in the university where QoEWA is applied) to take the decision to improve factor F1 (performance) by deploying a recent version of the Oracle WebLogic server. Accordingly, after launching the new version into live operation, the number of complaints raised by users against the performance was dramatically reduced (from 160 remedies to 75). This confirms that QoEWA with MOO functionality (Marler & Arora 2004) has the ability to identify and optimise the objective factors needed to improve QoE. However, this conclusion is subject to where and how QoEWA is applied, business requirements and technology.

Whilst MOO optimises the balance between KPIs and KQIs, it does not explain the rationale behind the gap between the two, i.e. MOO does not answer the question: Why do some users rate poor service as excellent in contrast to others who rate excellent service as bad? In spite of the results obtained in Iteration 1 confirming that the gap is small between KPIs and KQIs, there is still an implication that service quality does not always reflect users' levels of satisfaction and that users may be affected by other external factors, such as social network or WOM. Consequently, the lesson learned from this iteration led to a suggestion for developing another (Iteration 4) that aims to build an insight into QoE using the Agent-based modelling (ABM) approach for optimising QoE and examining user behaviour and interaction.

6.4 Summary

This chapter has presented an overview of the MOO approach, which has been employed in this research to evaluate and adjust the balance between the different objectives of QoE. MOO is used to control the balance between resources and user experience. That is, the existing design of QoEWA has extended by incorporating the MOO approach into the model to examine the trade-off between multiple objectives (KPIs and KQIs). The design has been utilised to build a simulation system that generates a set of points for establishing the non-dominated Pareto optimal solutions, which in fact demonstrates the optimal trade-off between KPIs and KQIs. The design has been implemented using the Netlogo tool, presented by Wilensky (1999). The instantiation includes two main functions: (1) The correlation function, which determines the correlation between KPIs and KQIs; (2) the optimisation function, which generates random values based on GA to produce the Pareto curve.

The developed Net Logo simulator imported the KPI and KQI values from the dataset produced by Iteration 1. Alongside these data, the simulator imports the predictive rules which are formulated in Iteration 2 by ML to predict the value of the KQIs. Accordingly, two tests have been performed. First, to validate the model by exploring the data and comparing it with the analysis obtained in Iteration 1 and second, to compute and predict the best optimal values.

Based on the test and evaluation, it has been found that MOO does not explain the rationale behind the gap between KPIs and KQIs, i.e. MOO does not answer the question: Why do some users rate poor service as excellent in contrast to others, who rate excellent service as bad? This leads to the extension of the design of QoEWA so as to study other external factors, such as social networks or WOM, as presented in the Chapter 7, where ABM is applied to elicit how these external factors influence QoE.

Chapter 7: The Insight and Perception of QoE

7.1 Overview

Based on the evaluation conducted in Iteration 3, this chapter incorporates an ABM approach into QoEWA to enable the ability to perceive user's QoE. This chapter is structured as follows: Section 7.2 presents an overview of the ABM approach adopted in this research to perceive and build an insight QoE. Section 7.3 presents the development of Iteration 4 through four sub sections that describes the design, instantiation, application and tests and evaluation. The tests are conducted through three scenarios, including exploration, what-if and agent interaction. Section 7.4 presents a summary of this chapter.

7.2 An Agent-Based Modelling Approach for perceiving QoE

Agent-Based Modelling (ABM) has been widely used (Rand & Rust 2011) as an approach for modelling and simulating the behaviour of complex systems, where elements and their behaviours interact with each other in a complex manner and dynamic environment (Macal & North 2007). It has been applied in many fields, such as computer science, complexity science, management science, social science, biology, business, economics, infrastructure and other scientific disciplines (Macal & North 2005). Ideally, it is applied when the behaviour of active objects or individuals is continually and dramatically altered over time and when the interaction between individuals with an environment is heterogeneous, complex, discrete, discontinuous and/or nonlinear (Bonabeau 2002).

ABM is appropriate and particularly useful for solving problems that require intelligent behaviour and adaptive learning, as well as those that involve coordination and strategic interaction to reflect a particular real-world situation and simulate behaviour over time from one life-cycle to another (Janssen 2005). However, ABM is not appropriate when the number of individuals is only one or two, i.e. other approaches such as game theory can provide a better modelling solution for small numbers of individuals (Macal & North 2007). Also, ABM becomes inefficient when the number of parameters, which define the characteristics are very large, which is because the design of the agent should consider the heterogeneity of the agent’s characteristics, rather than the description. The heterogeneity allows the system to track individuals and examine their behaviour over time, and hence, determine the factors that influence the behaviour (Bonabeau 2002; Rand & Rust 2011).

The term “agent” is generally defined as an independent component (e.g. model, individual, active object, software, etc.) that has certain characteristics represented by attributes, behavioural rules, resources, decision-making sophistication, memory and extended rules to modify the behavioural ones (Macal & North 2005). Fundamentally, an agent has to be developed in a way to maintain the following characteristics (Sargent 2014):

- **Autonomy:** An agent has to be self-directed, operating and interacting independently in its environment and with other agents;

- **Modularity:** An agent has to be self-contained and identifiable with a set of properties assessing its behaviour and decision-making capability;

- **Sociality:** An agent has to be social, having the capability of interacting with other agents within the environment to exchange knowledge;

- **Conditionality:** An agent has to have a state that changes over the time representing its condition, whereby the agent’s state is associated to its attributes and behaviour.

Each agent in the computerised ABM system behaves according to the behavioural rules, or the learning and experience obtained from other agents within the environment (Macal & North 2007). In terms of design, ABM like any other simulation modelling approach, has to have a specific well-defined purpose, input, output, measurements as well as mechanisms for performing verification and validation activities (Macal & North 2007; Sargent 2014). In order to construct a mature ABM that mimics real-life problem-solving situations, there are various steps normally followed (Twomey & Cadman 2002; Macal & North 2005; Rand & Rust 2011), as stated below:

1. Identify agents and the theory which describes their behaviour and characteristics;

2. Identify and describe the relationships between agents and how they interact to each other;

3. Define the environment and platform, where the agents are installed and interact;

4. Explore and inject agents with input data and assumptions;

5. Analyse and validate data collected from each individual deployed agent; and

6. Test and evaluate the overall performance of the developed ABM system.

In relation to the work here, it is observed from the literature that an ABM system has the ability to derive information and learn from agents and their environment through the senses, thereby gaining insight into an individual's perception and experience. Accordingly, much research effort has been put into examining the application of ABM in managing and evaluating customer experience and retention (Twomey & Cadman 2002; Hassouna & Arzoky 2011). Others have utilised ABM to simulate the influence of Word-Of-Mouth (WOM) communications, which can significantly influence behaviour (Hummel et al. 2012). From a service oriented point of view, ABM has been used by Puliafito et al. (1997) as a technique for managing QoS and evaluating users requirements, whereby agents are developed to represent their interactions with the provided services. From a QoE perspective, ABM has been deployed to evaluate users' perception and to optimise QoE accordingly (Gómez et al. 2013).

Building upon the above overview, this research involves applying ABM as a tool incorporated with QoEWA to assess user-perceived QoE of Web Application when resources are constrained (e.g. minimised, maximised or optimised). This allows for better insight and decision-making regarding QoE and user satisfaction as well the evaluation of KPIs and KQIs.

7.3 Iteration 4: User perception of QoE

7.3.1 Design of Iteration 4

In order to gain insight into the QoE assessment, the existing structure of QoEWA is extended to construct a heuristic design based on an ABM approach. This is because it is important to incorporate the evaluation of user satisfaction and brand loyalty into QoE optimisation processes as they are prohibitively expensive and can have detrimental consequences (Nokia 2004; Soldani et al. 2006). Consequently, a design decision was taken in Iteration 4 to examine the perceived QoE of Web Application as well as adjust and optimise QoE from the perspective of user satisfaction, loyalty and resource utilisation. The design here focuses on user behaviour, in contrast to that constructed in Iteration 3, which adjusted and optimised QoE exclusively from the perspective of resource utilisation.

Based on the problem defined in Subsection 3.4.1 and the suggested solution proposed in Subsection 3.4.2, it is found in the literature that ABM has the ability to provide insights regarding user perception of QoE (Twomey & Cadman 2002; Hassouna & Arzoky 2011; Gómez et al. 2013). Accordingly, an ABM model is developed here and structured into a set of sub-modules that interact with each other following the ABM architecture presented by Maes (1991). In addition, the design of ABM is described by the ODD protocol, which was proposed by Grimm et al. (2010) and has been widely used by agent-based modellers. The design includes three main components for describing the model: Overview, Design concept, and Details. These components are specified through nine stages that describe the model's purpose and design in sufficient depth: (1) define the purpose of ABM; (2) define the entities; (3) initialise the ABM environment; (4) identify the agents; (5) define process overview and scheduling; (6) specify design concepts; (7) define input; (8) set the initialisations; and (9) specify the sub-models.

- **Stage 1: Define the purpose of the ABM system**

The primary goal of developing the ABM system is to enable agents to participate as end-users; interacting with the environment, service and each other. Each agent is deigned to have its own behaviour based on the data obtained from Iteration 1 and the rules from Iteration 2. The ABM system explores and examines agents' behaviour under the what-if-scenario approach, which is usually used to evaluate how the simulator behaves under different situations (Twomey & Cadman 2002). Based on the solution suggested in Subsection 3.4.2, the following three main scenarios are simulated here:

- First scenario: Evaluates how well the developed ABM corresponds to the design and adapts to the reality, with consideration of the guidelines presented in Section 7.2;

- Second scenario: Examines user behaviour when: (1) Resources are limited to a minimum; (2) when resources are stretched to the maximum; and (3) when resources are optimised to maximise the number of promotors;

- Third scenario: Examines users' interaction when users within the same location interact with each other through WOM communication.

- **Stage 2: Define entities**

According to Maes (1991), the architecture of the ABM system specifies a set of entities, including the agents, environment, and their interaction. During the interaction, there is a continual input and output between agents and the environment (Akkiraju et al. 1998). By combining these two approaches, the design here is thus consists of three entities: Agent (represents end-user); environment (provides Web Application); and their interaction (includes inputs and outputs). These entities are structured as shown in Figure 7-1.

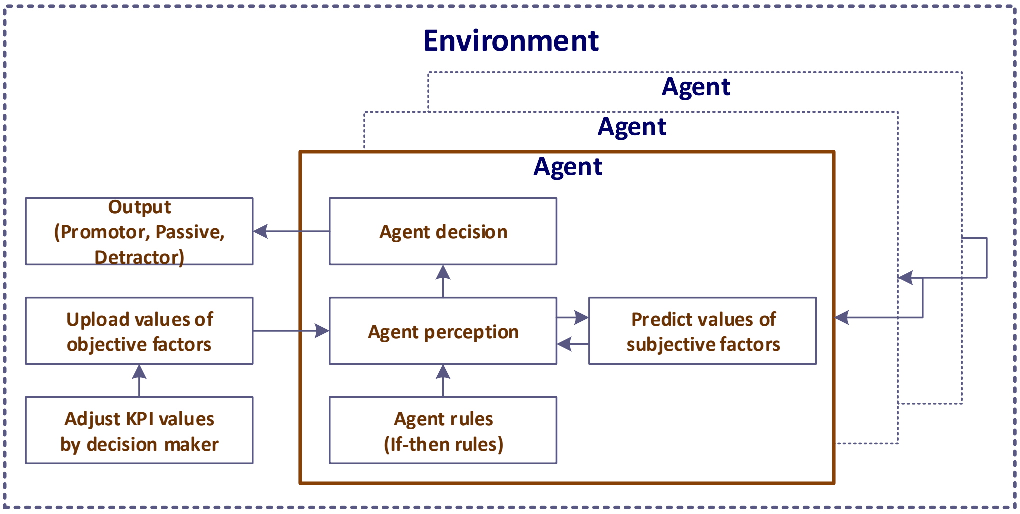


Figure 7-1: The proposed ABM Architecture

- **Stage 3: Initialise the ABM environment**

The environment is built with the consideration of modularity, following the model proposed by Gilbert & Terna (2000), which allows the modeller to simplify the code design by determining agent behaviour through external objects linked to rule maker ones. In the context of QoEWA, the environment is represented by the Web Application and the service platform, including the database, middleware and infrastructure. It provides information related to service quality to the agent, such as objective data obtained from the technical and operational dataset and assumptions set by the decision maker (e.g. minimise or maximise the KPIs). Then after processing, the decision maker in the environment entity is provided by the agent with information regarding the agent type, KPIs and KQIs. This enables the decision maker to decide which KPI value is most appropriate for implementing the strategy.

- **Stage 4: Identify the Agents**

The agent represents the end-user and interacts with the environment and other agents. Each agent has properties based on user profile and a behaviour based on constraints and rules (presented in Iteration 2 as ML rules). The agent is classified into three types (as shown in Figure 7-2), depending on its characteristics, measured by KPIs and KQIs, each of which is coloured green, yellow or red. This is determined by the following if-statements:

- *IF KPI < KQI THEN Agent = Green.* This indicates best optimal value of experience with fewer resources used;
- *IF KPI < KQI THEN AGENT = Red.* This indicates poor optimal value of experience with more resources used;
- *IF KPI = KQI THEN AGENT = Yellow*. This indicates average optimal value of experience with average resources used.

```
to assign-color
IF   v_QoE_Obj < v_QoE_Sub  [set color green  set shape "face happy"    set size 1 ]
IF   v_QoE_Obj = v_QoE_Sub  [set color yellow set shape "face neutral"  set size 1 ]
IF   v_QoE_Obj > v_QoE_Sub  [set color red    set shape "face sad"      set size 1 ]
end
```

Figure 7-2: Classification function

The rationale behind the classification above is derived from the view that there is a link between satisfaction and brand loyalty (Reichheld 2003). Thus, the classification of the agent can be determined by either: (1) The behaviour characteristics as expressed by satisfaction' or (2) an input from another agent that influences the behaviour (Akkiraju et al. 1998), i.e. the agent could be influenced by Word of Mouth (WOM) communication, as mentioned earlier in Rand & Rust (2011). Consequently, this is modelled here by utilising the Net Promotor Score

(NPS) approach, which was initially presented by Reichheld (1996) and then widely used by internet and network service providers as a benchmark to measure user satisfaction and loyalty. It classifies customers into three categories: Promotors, Passives, and Detractors. These categorise are incorporated with the MOS scale into QoEWA. The classification is carried out by asking customers about their satisfaction and whether they would recommend the service to others, with the answer being scored on a scale of 0 to 10:

- **Promotors (score 9-10):** Are satisfied with the services provided. They will recommend the service positively to others;
- **Passives (score 7-8):** Are not enthusiastic enough about the service provided to actually promote it or recommend it to others;
- **Detractors (score 0- 6):** Are not satisfied with the service provided. They can negatively affect the brand through WOM communication.

The correlation between KPIs and KQIs formulated by the ABM system establishes the NPS classification, (in which KPIs evaluate objective factors and KQIs evaluate subjective ones). The mapping between KPIs and KQIs classifies whether the user is a Promotor, Passive, or a Detractor. This is illustrated in Figure 7-3.

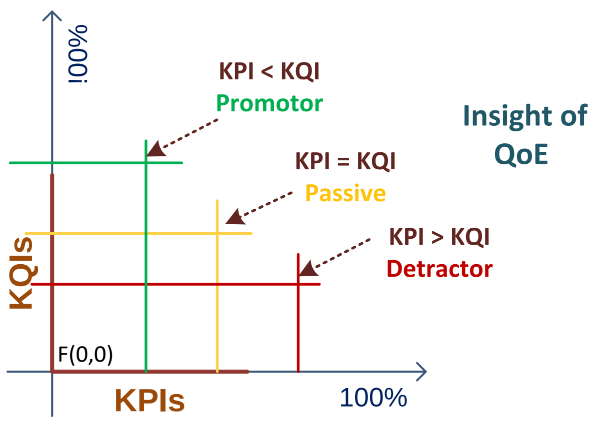


Figure 7-3: NPS classification for perceiving QoE

In relation to the MOS scale used in Iteration 1 and the obtained results, it is found that there is a significant link between this scale and that for NPS, as shown in Figure 7-4. The results confirm that the number of users who are classified as Promotors is fairly close to the number who have KPI less than KQI, likewise Detractors and Passives, whereby KPI is greater than KQI for the former, and KPI equals KQI for the latter

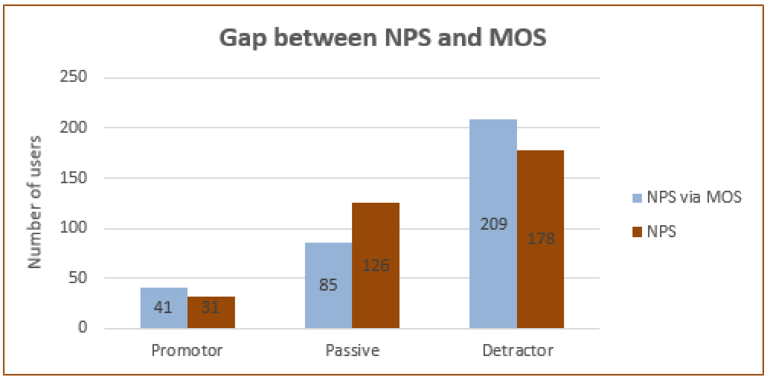


Figure 7-4: Comparison between the MOS and NPS scales

Hence, by mapping the MOS with NPS scales, as shown in Table 7-1, the developed ABM system will be able to perform the following two functions: (1) Using MOS to measure user satisfaction and to determine behaviour regarding the provided service; (2) using NPS to formalise the relationship and interaction between agents as well as studying how agents within the same location influence each other via WOM communications.

Table 7-1: Mapping between NPS scale and MOS Scale

MOS Model (ITU-T 2006b)		NPS Model (Reichheld 2003)		QoEWA		
Scale	**Quality**	**Scale**	**User type**	**Scale**	**QoE**	**Colour**
1	Bad	10	Detractor	10	KPI < KQI	Red Agent
1	Bad	20	Detractor	20	KPI < KQI	Red Agent
2	Poor	30	Detractor	30	KPI < KQI	Red Agent
2	Poor	40	Detractor	40	KPI < KQI	Red Agent
3	Fair	50	Detractor	50	KPI < KQI	Red Agent
3	Fair	60	Detractor	60	KPI < KQI	Red Agent
4	Good	70	Passive	70	KPI = KQI	Yellow Agent
4	Good	80	Passive	80	KPI > KQI	Yellow Agent
5	Excellent	90	Promotor	90	KPI > KQI	Green Agent
5	Excellent	100	Promotor	100	KPI > KQI	Green Agent

- **Stage 5: Define process overview and scheduling**

As pointed out in the introduction of this chapter, the ultimate goal of an ABM system is to gain insight into QoE assessment, thus for connecting this characteristic to the agent's duties, it should have in its memory rules that: (1) Calculate and predict its KPI and KQI values; (2) govern its behaviour and interaction; and (3) update its state and environment, e.g. update the decision maker with alternatives. Figure 7-5 shows an interaction diagram that illustrates the agent interaction with its environment.

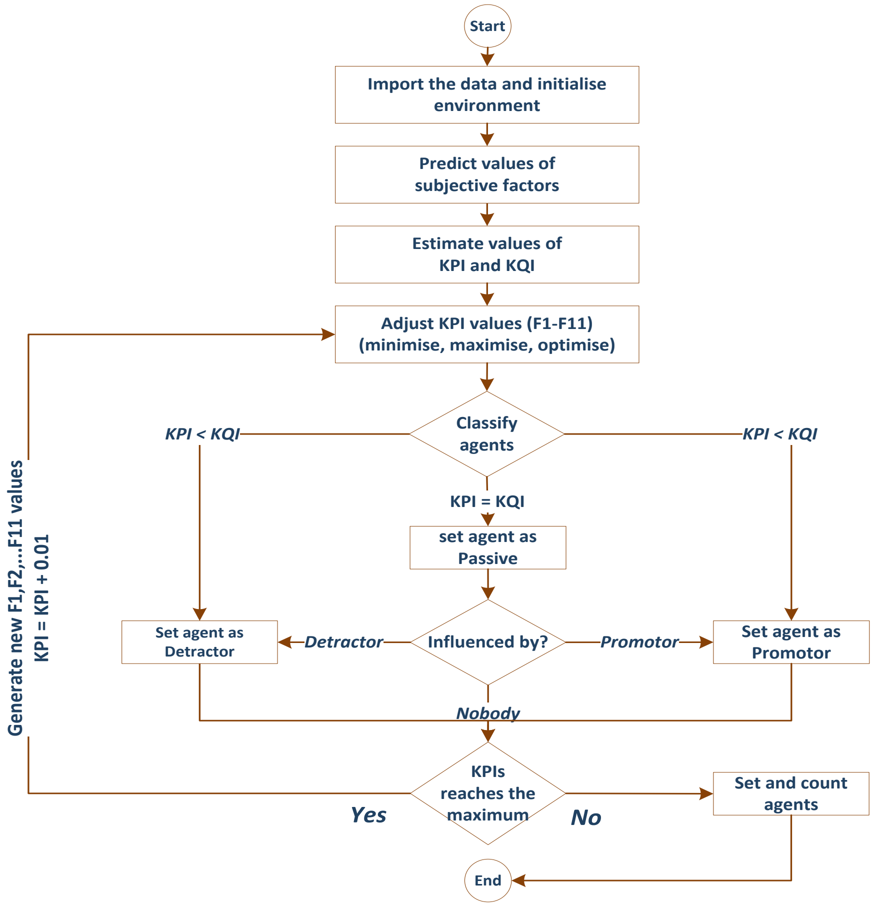


Figure 7-5: An interaction model for the proposed ABM

- **Step 1:** The ABM system imports the dataset, which includes objective data and information about users, such as location and application. The system then initialises the environment accordingly.

- **Step 2:** The system predicts KQI values based on the rules defined in Iteration 2 and the imported KPI data. Thereafter, each agent is explored in the environment with KPI and KQI values.

- **Step 3:** The assessor adjusts the KPI values and applies the if-scenario (maximise, minimise, optimise), where the KPI values are randomly generated based on the range of minimum and maximum KPI.

- **Step 4:** The system classifies the agent (Promotor, Passive and Detractor). In the Passive case, the system determines whether there is a chance of influencing in such a way as to turn the agent into either a Promotor or Detractor.

- **Step 5:** The system repeats the loop until the condition becomes false. The KPI values are generated based on the GA developed in Iteration 3, which enables the model to generate thousands of random possible values of KPI.

- **Stage 6: Specify the design concept**

The design of the ABM is based on a combination of different approaches towards the identified problem. This subsection summarises the concept of the design adopted. First, the architecture of the ABM is based on the approaches presented by Maes (1991) Akkiraju et al. (1998) for efficiently specifying the entities and their interactions, as well as describing the input and output between the agents and the environment. Second, mapping between the MOS (ITU-T 2006b) and NPS (Reichheld 2003) scales enables the developed ABM system to determine an agent's behaviour and interaction through user satisfaction and WOM communication. Third, the agent uses the rules derived from the ML approach to predict the KQI values. This enables the agent to have an adaptive behaviour and learn from experience. Fourth, the design is expected to be verified and tested rigorously using the guidelines proposed by Rand & Rust (2011) and the taxonomy of an ABM presented in Windrum et al. (2007). In sum, with the combination of the approaches adopted, the design has the capability of presenting the QoE problem in the context of ABM, which enables the decision maker to gain insight into how it can be solved.

- **Stage 7: Define input data**

The developed ABM system consists of a dynamic dataset that is imported from a flat file to the system. These data are analysed and prepared through a statistical and predictive process, as illustrated in Figure 7-6. The raw dataset has been extracted from a real application domain (as discussed in the previous chapters) and was statistically analysed in Iteration 1. Then, the data was processed by the Weka tool in Iteration 2 to generate predictive rules that determine the subjective values. Hence, the data extracted from Iteration 1 has been used as input from the environment to the agents to measure the KPI values, whereas those extracted from Iteration 2 have been used as rules for the agents to predict the KQI values. Thereafter, the system produced simulated data that is validated with the original real data.

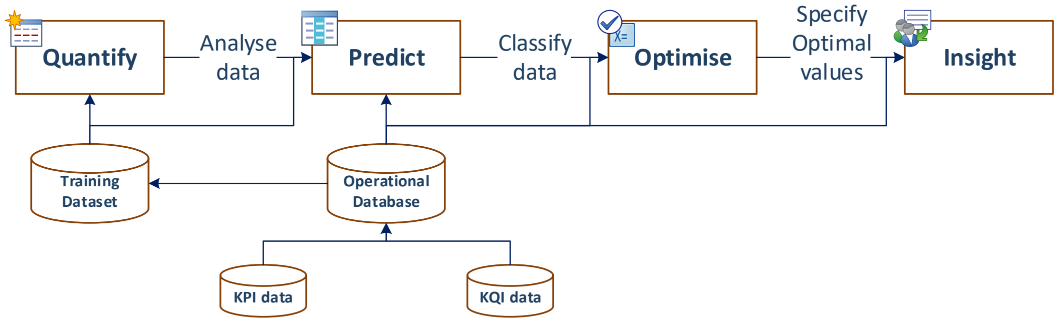


Figure 7-6: Preparing and processing the data for the proposed ABM

- **Stage 8: Set Initialisations**

To control the ABM system dynamically, the input parameters are associated with those that define the original dataset (which was used in Iteration 1). The value of each parameter is computed statistically by the ABM system and dynamically initialised as follows:

- The number of agents imported to the system, initialised by 335 agents;

- Initial number of Promotors, Passives and Detractors;
- The MOS scale (ranging from 1-5);
- The increment value of KPI, set as (0.01);
- The maximum, minimum, and average values of KPI are based on the analysis conducted in Iteration 1. The range generated by the ABM system for each factor is presented in Figure 7-7;
- The KPI value and the rate of Passives being influenced by Promotors is 76, i.e. 85% of Passives are most likely influenced by Promotors when the KPI value is greater than 76);
- The KPI value and the rate of Passives influenced by Detractors is 50, i.e. 80% of Passives are most likely influenced by Promotors when the KPI value is less than 50).

For determining the what-if scenario, the parameters above can be adjusted and re-initialised with new values depending on the tests to be conducted by the decision-maker.

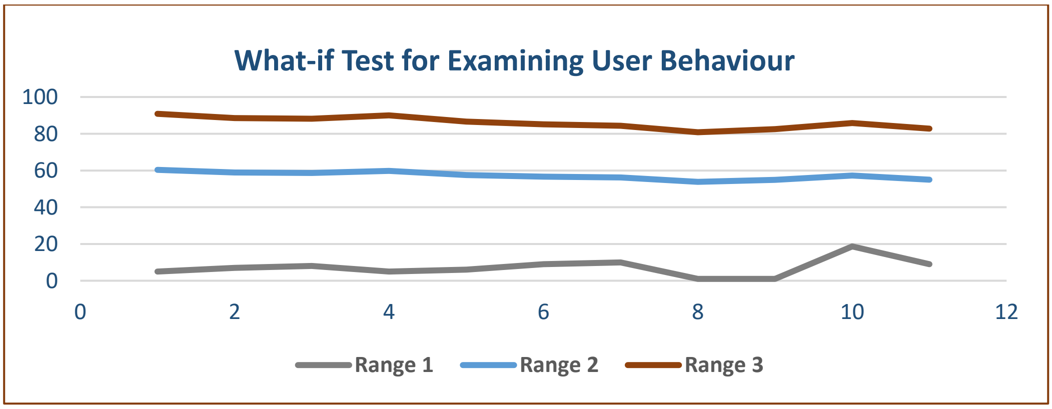


Figure 7-7: Range of values of the KPIs generated by ABM

- **Stage 9: Specify the sub-models**

The design encompasses three sub-models representing the context discussed above. The first is designed to explore and initialise the data, whilst the second demonstrates the behaviour when KPI is set with minimum, maximum and optimal values. The third mode examines the behaviour when users within the same location interact with each other. These sub-models are illustrated in more detail in the implementation and test subsections.

7.3.2 Instantiation of Iteration 4

The instantiation activity of this iteration is performed by NetLogo, which was utilised in Iteration 3 as a tool for the MOO problem. In terms of ABM, Netlogo is ideal tool for modelling complex systems developing over time. This is because it consists of a set of libraries and a programming language that allow the modeller to declare an agent's attributes and interactions, setup the model environment and to provide graphical application (Tisue & Wilensky 2004). Moreover, it comes with extension facilities that allow Netlego to be extended and/or integrated with other tools. For example, this research study involves using the CSV-Extension, which adds CSV parsing capabilities to the code and allows the NetLogo model to be dynamically integrated with output obtained from Iteration 1 and Iteration 2.

The ABM system can be implemented differently, however, since the methodology of the implementation is performed with an "agile" development process, which is divided into the following components.

- **GUI Component:** It is a form-based interface inherited from Iteration 3. It includes additional objects that control the ABM system functions. Figure 7-8 shows the buttons that invoke the processes used to examine an agent's behaviour and interaction. The "world" space element shows how agents (Promotors, Passives and Detractors) behave and interact by using colours (green, yellow, and red) to categorise them.

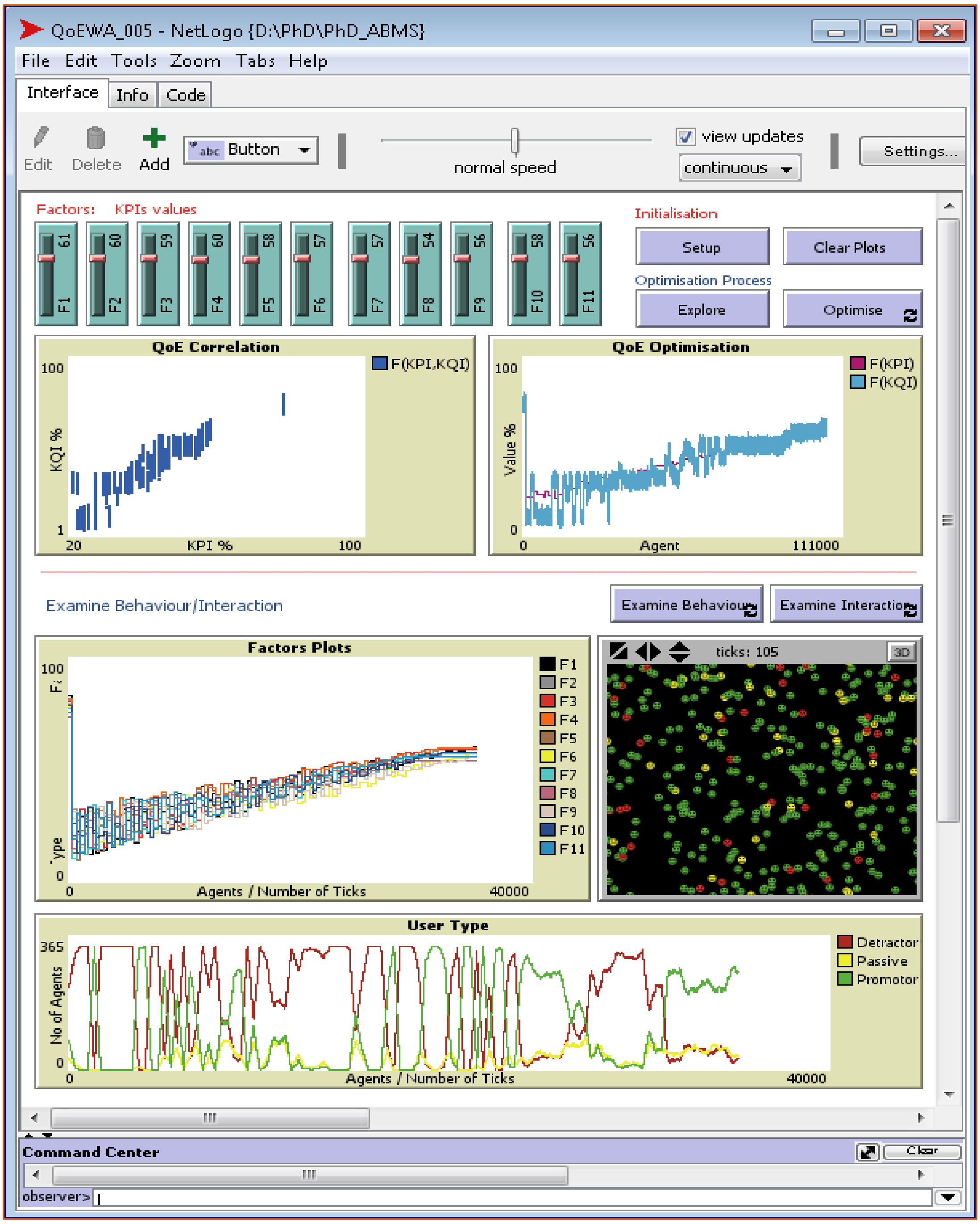


Figure 7-8: QoEWA Agent-based System

- **Data Component:** Since the input data used in this iteration are the same as those used in Iteration 3, this component is derived from the latter to process the input of the KPI and KQI data. However, this component has additional input parameters for initialising the rate of Passives influenced by Promoters and Detractors as well as having parameters for the incremental values of KPI.

- **Output Component:** In addition to the plots described in Iteration 3, this component includes: Factor plot and a user type plot. The former presents the random values of the objective factors (F1-F11) generated by the system, while the latter provides the number of agents for each category. During the process, the system generates the values of the KPIs and KQIs in the command centre.
- **Configuration Component:** It has two main procedures: (1) “setup”, which builds and initialises the world that the agents will interact with and (2) “go”, which is executed repeatedly with an infinite loop to generate agents until the condition becomes false, i.e. when the generated value of each objective factor corresponds to the maximum.
- **ABM Engine Component:** It includes the core functions and the business logic of the ABM system, which examine an agent’s behaviour and interaction. The input of this component is provided by the input and configuration component. While the output is presented by the output component, which presents the values of the KPIs and KQIs. The sliders in the interface can be used to examine the scenarios defined earlier in the design.

7.3.3 Application and Testing of Iteration 4

The test is rigorously verified and validated through the guidelines proposed by Rand & Rust (2011), and the taxonomy of the ABM presented by Windrum et al. (2007). This combination allows for each test case to be independently performed under the consideration of several factors, including: (1) The nature of the object being tested; (2) sensitivity analysis with respect to the obtained optimal values; and (3) verification and validation. Consequently, by reflecting the design concept described above, the developed ABM system is tested in three scenarios based on those defined in Stage 1 of Subsection 7.3.1: The first scenario presents the process which initiates and explores the current situation. The second presents a what-if scenario and examines an agent's behaviour, whilst the third examines its behaviour and interaction.

- **Test 1: Exploration**

This test examines the scenario where the developed ABM system imports the objective data of end-users and explores them randomly in the space in which users are represented by agents. The test then evaluates and compares the values that the agents initialised with the real values used in Iteration 1. This is to ascertain how well the developed model corresponds to the design and adapts to the reality, with consideration to the guidelines presented by Rand & Rust (2011). On the other hand, in order to maximise the reliability of the developed ABM system, the test follows the replication model presented by Zhong & Kim (2010). Thus, the exploration process is repeated and invoked three times on the same simulated dataset, but with new generated subjective values obtained from the prediction rules generated by the ML model (developed in Iteration 3).

In the experiment, the results obtained from each individually performed run show very little variation with the number of agents of each type, as shown in Figure 7-9. As aforementioned, in each run, the system imports 335 agents and the average numbers of Promotors, Passives, and Detractors 55, 7, and 272, respectively. These results are very close to the numbers obtained from the real dataset, namely, 57, 12 and 268) respectively. This actually confirms the validity of the developed prediction methods used for measuring the values of the KQIs.

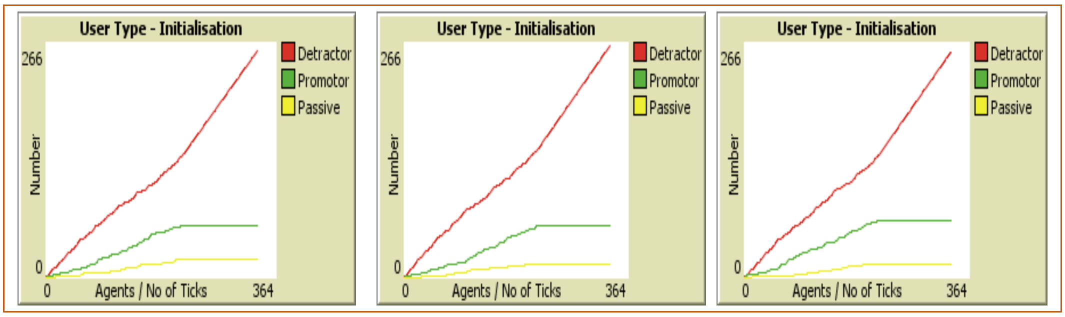


Figure 7-9: The number of agents of each type

In addition to examining the sensitivity of the ABM output, the above run is repeated but with different pool-sizes to verify the reliability of the model and to confirm that any further increase in the number of agents will not affect the ABM results. Hence, the pool size is duplicated and set to 335, 670, 1340 for the first, second, and third runs, respectively. The results confirm that the average number of Promotors, Passives, and Detractors is 55, 7, and 272 for each run, respectively. Hence, this consistency confirms the sensitivity and reliability of the model.

- **Test 2: What-if Test for Examining User Behaviour**

This test is based on the initialisation and creations generated by the previous test (Test 1). It is engineered to simulate and examine agents' behaviour (i.e. it determines whether it is a Promotor, Passive, or a Detractor) under the following scenarios: (a) When resources are limited to the minimum; (b) when resources are stretched to the maximum; and (c) when resources are optimised to maximise the number of promotors.

a) Examination of user behaviour when the KPIs are limited to the minimum

In order to carry out this test, the KPI values are set to the minimum limit, i.e. this is obtained from the statistical analysis conducted in Iteration 1 and listed in Table 3-2 in Chapter 3. Thereafter, the behaviour examination process is invoked to initialise the agents, which establishes the range of the minimal resources. Figure 7-10 illustrates at the top left-hand side, the variation of each factor, which ranges from the minimum value to the average.

Figure 7-10 demonstrates at the right-hand side the spaces of the agents and how their behaviour is changed accordingly. The plot at the bottom is updated when they are created with the number of each agent's type, i.e. whether it is a Promotor (green), Passive (yellow), or a Detractor (red).

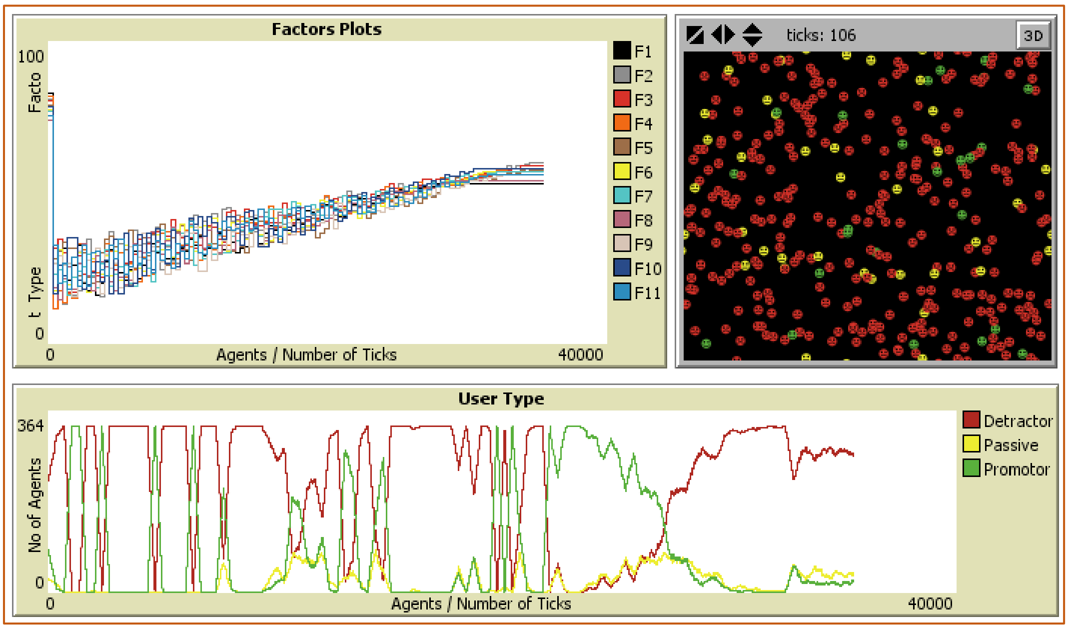


Figure 7-10: Behaviour when the KPIs are limited to the minimum

The system generates 35,510 ticks based on the created agents (i.e. the tick-counter is incremented each time the simulation cycle/run is repeated). Each cycle generates a new set of agents based on the initialised set (i.e. 335 agents). The system continues generating new ticks incrementally from the minimum values until reaching the average value of the objective factors and the results are generated as shown in Figure 7-11. The best results are obtained when the number of promotors (ticks) reaches the maximum peak. It emerges that approximately 33% of the agents are set as promotors when the values of the KPIs are generated between Range 1 and Range 2 shown earlier in Figure 7-7.

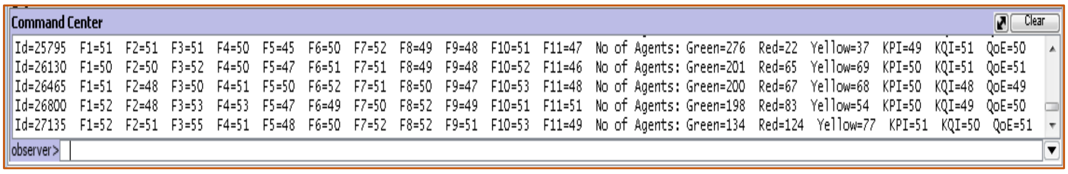


Figure 7-11: The generated optimal values

b) Examination of user behaviour when the KPIs are set to the maximum

Unlike the previous test, here, the KPIs are maximised along with the values calculated from the statistical analysis conducted in Iteration 1. Thereafter, the behaviour is examined by invoking the examination process, which establishes the range of the maximum resources. Figure 7-12 illustrates at the top left-hand side the variation of each factor, which ranges from the average values up to the maximum.

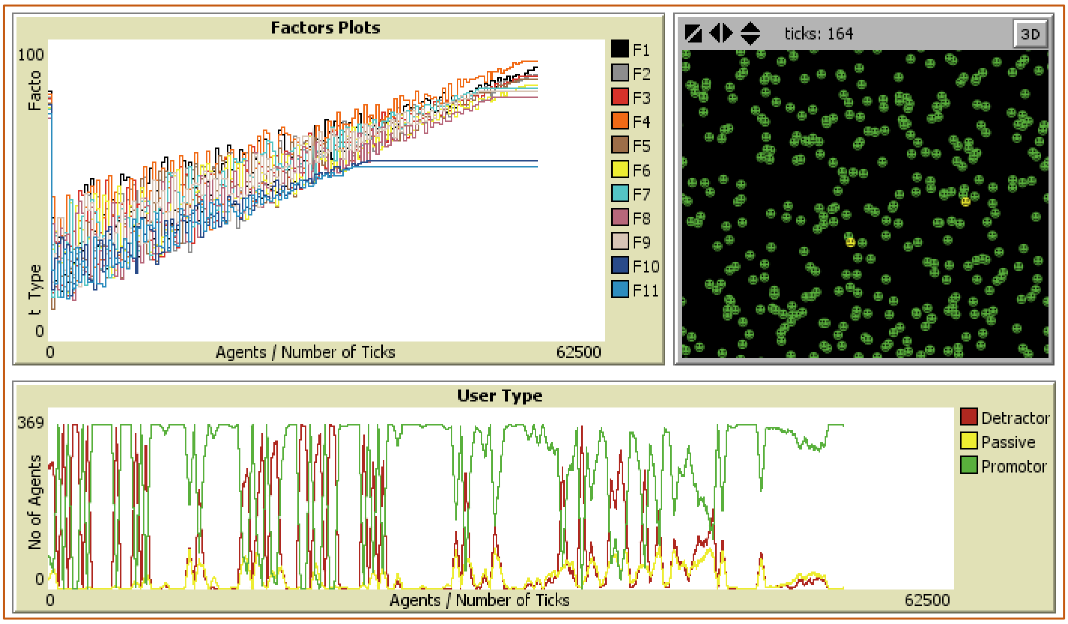


Figure 7-12: Behaviour when the KPIs are set to the maximum

The system generates 54,711 ticks based on the created agents. It keeps generating new ticks incrementally from the average values until reaching the maximum value of the objective factors and the results generated as shown in Figure 7-13. Similar to the previous run which minimised values, when the KPIs are limited to the maximum, the best results are obtained when the number of Promotors (ticks) reaches the maximum peak. It is found that approximately 70% of the agents are set as Promotors when the values of KPIs are generated between Range 2 and Range 3 shown earlier in Figure 7-7.

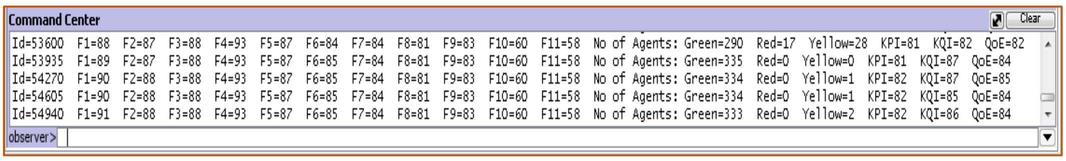


Figure 7-13: The generated optimal values

c) Examination of user behaviour when resources are optimised to maximise the number of promotors

In contrast to the above two runs, which are configured to minimise and maximise the KPI, this run is conducted without setting the KPI values. This is because the system aims to determine all the optimal values that maximise the number of Promotors with minimal technical resources. Figure 7-14 illustrates at the top left-hand side the variation of each factor, which ranges between the minimum and maximum values of the KPIs.

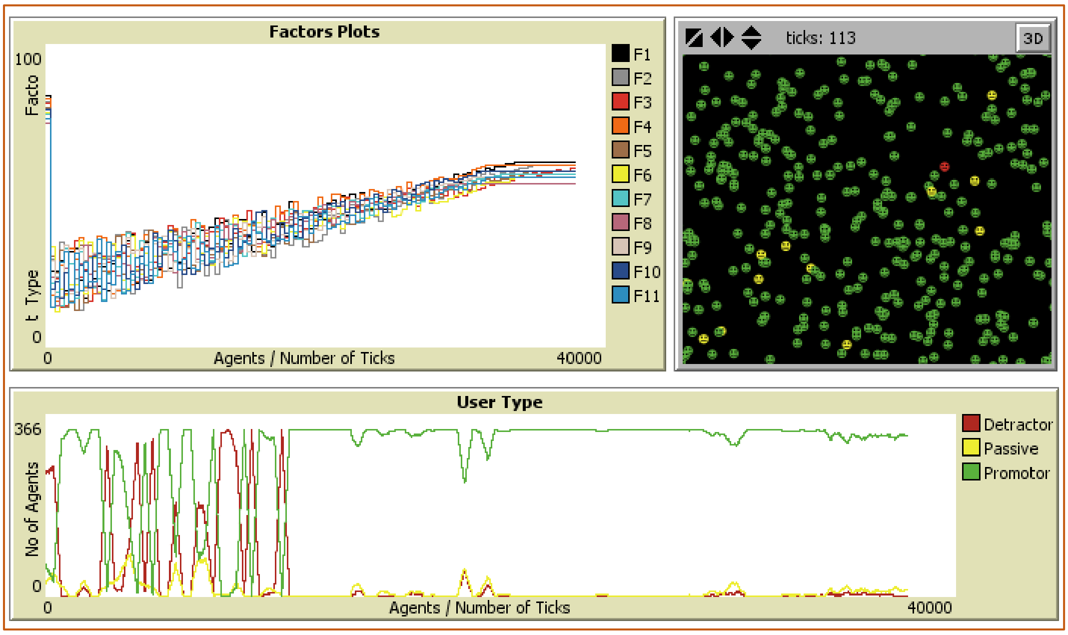


Figure 7-14: Behaviour when resources are optimised to maximise Promotors

The system generates 37,855 ticks based on the created agents. The system keeps generating new ticks incrementally from the minimum values until reaching the maximum value of the objective factors (i.e. the target of this test is to generate the maximum number of possibilities so as to maximise the number of promotors) and the results are generated, as shown in Figure 7-15. It emerges that approximately 80% of the agents are set as Promotors when the values of KPIs are generated between Range 1 and Range 3 shown earlier in Figure 7-7. It can be concluded that the most obvious factors that influence the percentage of Promotors are F1 and F4.

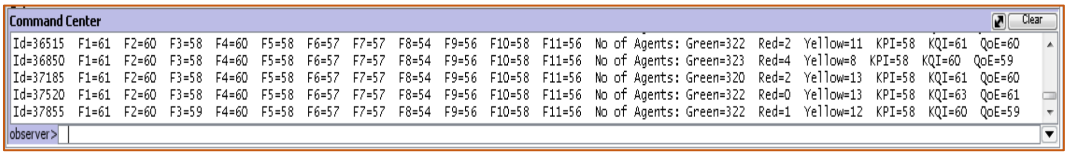


Figure 7-15: The generated optimal values

The results obtained from the three what-if tests conducted above are summarised in Table 7-2.

Table 7-2: Summary of the preformed what-if tests

What-if Scenario	Range of KPI's values generated by the ABM	Number of Ticks	Percentage of the Promotors
A. KPIs are limited to minimum	From Range 1 to Range 2	35,510	33%
B. KPIs are set to maximum	From Range 2 to Range 3	54,711	70%
C. KPIs are optimised	From Range 1 to Range 3	37,855	80%

- **Test 3: Examine User Interaction**

This test is based on the previous one, which examines the behaviour when resources are optimised to maximise the number of Promotors. Hence, during the optimisation process, this test determines the possibilities of converting Passive agents into either Promotors or Detractors. Passives are converted in this test because they are associated with an unstable situation, unlike the counterparts (i.e. Promotors and Detractors), who are defined as active users and associated with a stable situation (Strandberg et al. 2011). It is important to ensure that the agents behave in a stable manner with respect to their goal (Zagorecki et al. 2010). This narrows the decision-criteria to two possibilities (Promotors or Detractors) and as a result allows the ABM system to produce stable probabilities, with less variation in decision alternatives. According to the analysis conducted on the dataset used in Iteration1, users who work within the same location (department) are most likely classified as Promotors when the KPI value is between 74% and 99% and

Detractors when this value is less than 40% whilst Passives are detected when it is between 40% and 74%. Taking this value as an input to the conversion process, Passives are converted to Promotors or Detractors, depending on the KPI value and location. This conversation process is illustrated in Figure 7-16, where the target Passive is linked by lines with Promotors or Detractors and its colour changed accordingly.

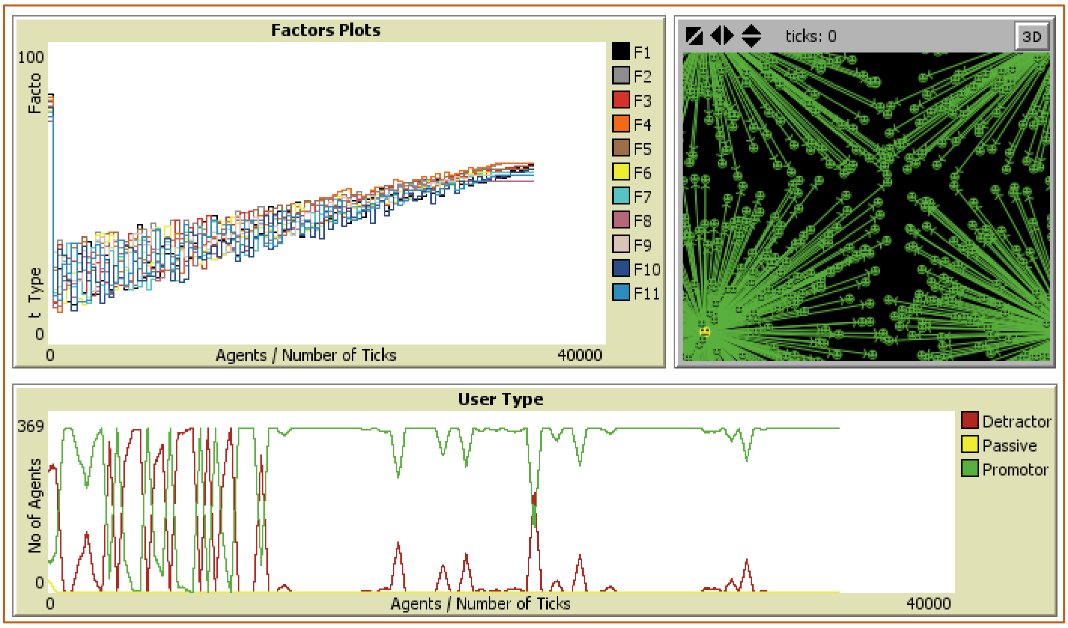


Figure 7-16: User interaction test

The system runs the optimisation and the converting processes repeatedly in a loop until the total number of Detractors goes to zero. It is found that there are 981 Passives out of 36,515 who are converted to Promotors and 442 to Detractors. Figure 7-17 shows the number of the influenced Passives.

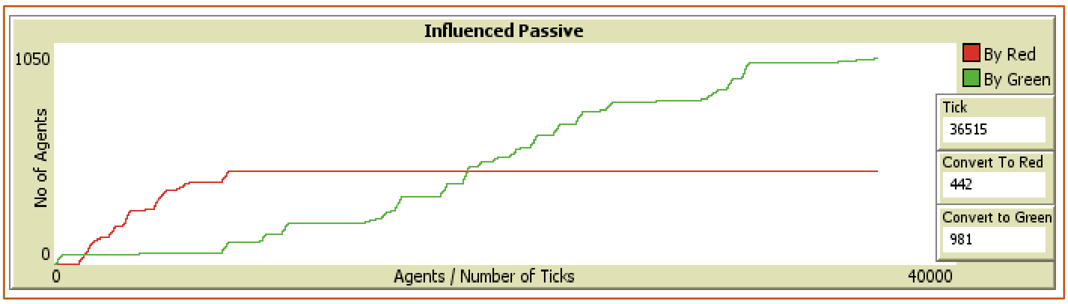


Figure 7-17: Number of Passives converted to Promotors and Detractors

7.3.4 Evaluation of Iteration 4

QoEWA with ABM functionality has the ability to provide sufficient insight into QoE by determining how given users behave in a particular situation, i.e. how they become Promoters, Passives or Detractors. In addition, the ABM system enables the decision maker to understand the factors behind QoE, and adjust the values of the KPIs in a way that improve KQIs, thereby increasing the number of Promotors and decreasing that of Detractors. Incorporating the MOS and NPS approaches into QoEWA facilitates the ABM system in: (1) Determining user satisfaction and behaviour regarding the provided service; and (2) uncovering the relationship and interaction between users, in particular, how users within the same location can be influenced by each other via WOM communication.

Since the aim of this iteration is to gain insight into QoE, the proposed ABM is designed to understand the objective and subjective factors behind it and to determine how the KPIs and KQIs can be adjusted so as to increase the number of Promotors and Decrease that of detractors. The results obtained from the exploration scenario have been compared and validated with the results obtained by Iteration 1 (which is based on a real-life dataset). The results obtained from the what-if scenario have been validated by those obtained from Iteration 1 and Iteration 3. The results of both, i.e. the exploration and what-if scenarios, are very close to the results of Iteration 1. This indicates the efficacy of the ABM system as the results are based on a real dataset. Moreover, the results obtained from the ABM system show that users work who within the same department and at the same location are most likely to be influenced by each other positively or negatively, despite whether the quality of the provided service is good or bad. The results obtained from the interaction scenario are validated based on the rules defined by the analysis, i.e. it emerged that users who work within the same location are most likely to be classified as Promotors when the KPI value is between 74% and 99%, and classified as Detractors when this value is less than 40%. Despite of the advantage of incorporating the ABM approach into QoEWA, the developed agent in the latter contains only a few socio-demographic characteristics, such as location and satisfaction. Hence, an ABM system could be more efficient, if other socio-demographic characteristics (such as age, gender, ethnicity, education, occupation, amongst others) are considered for examining user behaviour and interaction. However, the effect of socio-demographic characteristics, such as age, gender, ethnicity and occupation, are beyond the scope of this research, due to the regulations regarding data protection.

7.4 Summary

This chapter has provided an overview of the ABM approach by explaining the process of building an ABM system. ABM has subsequently been applied as a tool incorporated with QoEWA to assess user-perceived QoE of Web Application when resources are constrained (minimum, maximum, or optimum). This allowed for better insight and decision-making towards QoE and user satisfaction as well as the evaluation of the KPIs and KQIs. The design of ABM is described by the ODD protocol, as proposed by Grimm et al. (2010) and is widely used by agent-based modellers. The design includes three main components for describing the model: Overview, Design concept, and Details. These components encompass seven sub-components that describe the model’s purpose and design in sufficient depth: (1) Purpose, (2) Entities, State Variables and Scales, (3) Process Overview and Scheduling, (4) Design Concepts, (5) Input, (6) Initialisation, and (7) Sub-models. The instantiation activity of developing the ABM has been performed by the NetLogo tool. The system is based on the following components: GUI, data processing, output/input, configuration and ABM engine. The testing of the ABM system has been rigorously verified and validated through the guidelines proposed by Rand & Rust (2011) and the taxonomy of the ABM presented by Windrum et al. (2007). The developed ABM system has been tested in three scenarios. The first is with regards to the process that initiates and explores the current situation. The second pertains to a what-if scenario, examining agent’s behaviour, whilst the third probes agents’ behaviour and interaction.

Chapter 8: Overall Evaluation and Conclusion

8.1 Overview

This chapter delineates the key conclusions and contributions of this research as well as highlighting the limitations to establish the potential direction for future research. This chapter is structured as follows: Section 8.2 presents a summary of this research providing a brief evaluation of each chapter. Section 8.3 identifies the research contributions to the knowledge-based IS research and environment. Section 8.4 provides an overall evaluation of how well the DSR approach was applied. Section 8.5 is focused on how well the defined objectives have been achieved. 8.6 discusses the limitations and possible opportunities for future research.

8.2 Summary of the Research and Findings

This research introduced a model called QoEWA for assessing the QoE of Web Application. QoEWA was developed to overcome the limitations discussed in Chapter 2 and Chapter 3, in which the development of QoEWA was framed by a DSR approach. QoEWA was developed through four iterations presented in four chapters: Chapter 4, Chapter 5, Chapter 6 and Chapter 7. The chapters of this research are summarised as follows.

Chapter 2: Presented a comprehensive introduction to the Web Application, along with a review of the standard software quality models. It was obvious that most of these models, which have been derived from standards, such ISO-9126 and ISO-25010, ISO 9241-11, TOGAF, and OASIS or from approaches, such as SOA and (Ramler et al. 2002; Calero et al. 2005; Malak et al. 2004; Zeng et al. 2007) do not address the fundamental issues that concern the quality of Web Application from a user perspective. In addition, they do not consider the complementary relationship between what is offered and what is expected. Consequently, this chapter investigated the QoE approach, which has traditionally been adopted for determining the correlation between quality of service and user satisfaction (Offutt 2002a; Cecchet et al. 2013; Phillips et al. 2015). Finally, after reviewing the key elements of QoE, the chapter addressed the challenges in adopting QoE for Web Application, including the challenges of defining and mapping the objective and subjective factors, as well as those of quantifying, predicting, optimising and perceiving QoE.

Chapter 3: Presented an overview of the DSR approach and justified why it was utilised in this research. It was found that due to the multidisciplinary nature of QoE, DSR has the theory to study the IS characteristics of the objective and subjective factors, as well as the relationship between them. The chapter discussed the design process and the theory that guided the research development and its outputs, which included the conceptual and technological artefacts. Thereafter, it provided a framework that combines the popular process models and guidelines for DSR (Hevner et al. 2004; Kuechler & Vaishnavi 2008; Peffers et al. 2008) with the design theory covered by Jones & Gregor (2007). Consequently, the DSR approach was applied through five stages: The first stage defined the research problem and scoped the problem from an IS perspective; the second stage evaluated the available solutions; the third stage developed the artefacts through a build-evaluate cycle; the fourth stage evaluated the development; and the fifth stage provided a conclusion on the knowledge and experience obtained through the research life cycle. The development stage included four iterations for quantifying, predicting, optimising, and perceiving QoE,

Chapter 4: Presented an overview of Actual-Versus-Target approach which was employed in this research to quantify the relationship between the objective and subjective factors (determined by the KPIs and KQIs). A design decision was taken to develop Iteration 1 to: (1) Explore systematically the correlation between the KPI and KQI measurements; and (2) quantify QoE by making a comparison judgment between what is actually measured and what it is targeted. The design was then implemented following an agile development process so as to break the development's activities into set-by-step increments with minimal advanced planning. The developed QoEWA was deployed and tested twice in two scenarios to: (1) Provide a benchmark in relation to the state-of-art by examining the correlation between KPIs and KQIs; (2) run the data in the context of the full QoEWA model and quantify QoE by measuring the actual values versus the target values. The obtained results showed that there is strong correlation between KPIs and KQIs, as well as a small gap between the measurements of the two. However, it was found that only 60% of the users provided their feedback on the quality of the service, which it means tracking user satisfaction was not possible (Mitra et al. 2011). This led to extending the design to measure MOS intelligently using an ML.

Chapter 5: Presented an overview of the ML approach employed in this research for prediction. In literature, it was found that there are five common supervised learning algorithms including: Decision Tree (DT), Naive Bayes (NB), K-Nearest Neighbours (KNN), Sequential Minimal Optimisation (SMO) and Random Forest (RF). In this chapter, a design decision was taken to develop Iteration 2 and incorporate the five supervised algorithms into QoEWA, The performance and accuracy of each algorithm was tested and evaluated through the following common statistical tests: True Positive (TP); False Positive (FP); Precision; Recall; and the F-measure. The trade-off between TP and FP was determined by the evaluation of the ROC curve. The lesson learned from the evaluation is that: (1) The DT algorithm gives the best performance accuracy, while the SMO algorithm gives the worst results; (2) ML enables QoEWA to facilitate the relationship between the objective and subjective factors, as well as predict the unknown values of KQI, i.e. satisfaction; and (3) ML enables the QoE assessor to understand the links and requirements that bridge between service quality and user experience. However, ML alone does not provide a prescription for controlling and optimising QoE. Consequently, the MMO approach was applied in the next chapter to control and optimise the balance between KPI and KQI.

Chapter 6: Presented an overview of the MOO approach employed in this research to examine the relation between the perceived user experience and the technical resources and hence, evaluate the trade-off between multiple objectives (KPIs and KQIs). A design decision was taken in this chapter that Iteration 3 would incorporate the MOO approach into QoEWA. MOO is employed as a multi-criteria technique to determine the Pareto set, which is used in the design here as a curve that represents the trade-off between KPIs and KQIs. The Pareto set is generated by a simulator to determine three points (A, B, and C) for drawing the Pareto curve. At point A, resources are significantly less than at point B and C, hence quality is not the highest, but point A can be taken as the best alternative value in terms of user experience. While point C can be taken as the best alternative value, if the focus is on quality of service. The balance between user experience and resource is indicated by the balance between the KPIs and KQIs, which depends on user and business requirements. The Pareto set values are generated by a simulator that is based on GA (i.e. A, B, C and the values between them). The results obtained from the simulator were analysed and formulated in the context to identify the factors behind lower satisfaction ratings. It was found that factors (F1, F2, F10, and F11) are important for gaining better performance of QoE. This, as a result prompted the solution architect (in the university where QoEWA was developed and applied) to take the decision to improve factor F1 (performance).

Chapter 7: Presented an overview of the ABM approach which was employed in this research to gain better insights and decision-making regarding QoE and user satisfaction as well as for the evaluation of KPIs and KQIs. A design decision was taken in this chapter to develop Iteration 4. An ABM system was developed based on the ODD protocol proposed by Grimm et al. (2010). The design includes three main components for describing the model: Overview, design concept, and details. The design was implemented by using the NetLogo tool. The agent is classified into three types, depending on its characteristics, measured by

KPIs and KQIs, each of which is coloured green (Promotor), yellow (Passive), or red (Detractor). The developed ABM system is tested in three scenarios: The first scenario presents the process which initiates and explores the current situation; the second pertains to a what-if scenario and examines agent's behaviour; and the third examines agents' behaviour and interaction. The results obtained from the test confirmed the validity of the developed ABM model and determined the values of KPI which increase Promotors and decrease Detractors. In addition, it was found that users within the same department are influenced by each other positively or negatively, regardless whether the performance of the service is good or bad.

8.3 Research Conclusions and Contributions

The main contribution of this research is to provide an architectural framework framed by DSR to assess the QoE of Web Application. Specifically, there was the consideration to cover three aspects of contributions embodied by Hevner et al. (2004): (1) Contributions to IS research (artefacts); (2) contributions to the knowledge base; and (3) contributions to the environment.

8.3.1 Contributions to IS research (Artefacts)

Through this research two different types of artefacts have been developed, including the QoEWA (conceptual) model and its (technological) instantiation, each of which is created in two complementary phases: (1) A behavioural science phase that carries out the research through the development and justification of theories; (2) and a design science phase that carries out the research through the building and evaluation of artefacts. This allows for the incremental development of QoEWA through a build-evaluate cycle, starting from the artefacts which facilitates the relationship between KPIs and KQIs and ending up eventually with IS artefacts incorporated into QoEWA as utilities providing the following constructs, models, methods, and instantiations:

- **QoEWA Artefacts for Quantifying QoE**

A set of artefacts have been developed in this research to enable researchers and practitioners to gain theoretical and technical knowledge that connects QoE measurement theories (e.g. Alreshoodi & Woods 2013; Aroussi & Mellouk 2014) to the gap analysis technique to facilitate the QoE measurement by quantifying the relationship between the KPIs and KQIs. The developed artefacts are presented in Table 8-1.

Table 8-1: Summary of the artefacts that quantify QoE

Artefact Type	Outcome
Construct	Iteration 1 constructed a solution based on Actual-Versus-Target approach to the QoE quantification and measurement problem, which is addressed by Alreshoodi & Woods(2013) and Aroussi & Mellouk (2014) and presented in Chapter 1 and Chapter 2.
Model	Iteration 1 presented a model, as shown in Figure 4-1 that quantifies QoE by utilising the Actual-Versus-Target approach, which enables a more holistic measurement and bridges the gap between the actual measurements and the measurements defined by the Service Level Agreement (SLA). The ratio between actual and target measurements allows the QoE assessor to monitor QoE easily by dealing with quantified and measurable values.
Method	Iteration 1 provided a method that defines the solution process of quantifying QoE. The method is described in Figure 4-2.
Instantiation	Iteration 1 provided a computational instantiation of the proposed model, including a GUI application that is implemented by Oracle ADF technology (a screenshot is shown in Figure 4-5). The logical structure of the application is described by UML class model and UML sequence model in Figure 4-3 and Figure 4-4.

- **QoEWA Artefacts for Predicting QoE**

To predict QoE, a set of artefacts have been developed to enable researches and practitioners to gain theoretical knowledge and practical experience in the fundamental aspects of designing, building, and applying ML approach in the area of QoE of Web Application i.e. this allows QoE assessor to understand the links and requirements that bridge between the service quality and user experience. The developed artefacts are presented in Table 8-2.

Table 8-2: Summary of the artefacts that predict QoE

Artefact Type	Outcome
Construct	Iteration 2 constructed a solution based on ML approach to the Web QoE prediction problem, which is addressed by Menkovski, Liotta et al. (2009), Menkovski, Exarchakos & Liotta (2010), and Aroussi & Mellouk (2014) and presented in Chapter 1 and Chapter 2.
Model	Iteration 2 provided a model, as shown in Figure 4-2 that predicts QoE by utilising an ML approach, which enables the QoEWA model to predict and evaluate subjective data dynamically (via MOS), based on limited user data, in a manner that builds a training dataset intelligently from a few samples. Also, ML helps to determine the relationship between the objective and subjective factors, i.e. it enables the model to predict the unknown subjective value from the known objective one (Alreshoodi & Woods 2013). As a result, this facilitates the mapping between the KPIs and KQIs.
Method	Iteration 2 provided a method that defines the solution process of predicting QoE. The solution process includes the steps required to predict and classify the value of MOS of QoE.
Instantiation	Iteration 2 provided a computational instantiation with a WEKA interface, shown in Figure 5-3 that can be used by practitioners to create the rules needed for predicting the MOS values of QoE.

- **QoEWA Artefacts for Optimising QoE**

A set of artefacts have been developed to allow researchers and practitioners to gain knowledge that links between QoE optimisation (e.g. Baraković & Skorin-Kapov 2013) and MOO technique to practically optimise quality and control the balance between resources and user experience. The developed artefacts are presented in Table 8-3.

Table 8-3: Summary of the artefacts that optimise QoE

Artefact Type	Outcome
Construct	Iteration 3 constructed a solution based on the MOO approach to the QoE control and optimisation problem, which is addressed by Al-Moayed & Hollunder 2010, Sharma et al. 2012, Song et al. 2012, and Baraković & Skorin-Kapov 2013 and presented in Chapter 1 and Chapter 2.
Model	Iteration 3 provided a model, as shown in Figure 6-1 that optimises QoE by utilising the MOO approach, which enables the QoEWA model to adjust the balance between KPIs and KQIs with respect to resource available and user experience requirements. In addition, MOO enables the model to determine the relationship between perceived user experience and technical resources, thus resulting in the identification of the objective factors needed to improve QoE.
Method	Iteration 3 provided a method that defines the solution process of optimising and controlling QoE. The steps of the solution process is illustrated in Figure 6-2.
Instantiation	Iteration 3 provided a computational instantiation of the model with a NetLogo interface (shown in Figure 6-3) that can be used by practitioners to adjust the balance between network resources and customer satisfaction, as well as define the optimal KPI values.

- **QoEWA Artefacts for perceiving QoE**

A set of artefacts have been developed with a link to the artefacts developed in Iteration 2 to enable the ability to perceive user's QoE. An ABM is proposed as an analytical technique for QoE of Web application to enable researchers and practitioners to obtain knowledge and practical experience of developing and applying ABM approach in the area of QoE. The ABM artefacts are presented in Table 8-4.

Table 8-4: Summary of the artefacts that perceive QoE

Artefact Type	Outcome
Construct	Iteration 4 constructed a solution based on ABM to QoE perception problem, which is addressed by Hummel et al. 2012, Dusi et al. 2012, and Soldani et al. 2006 and presented in Chapter 1 and Chapter 2.
Model	Iteration 4 provided a model, as shown in Figure 7-3 that perceives user's QoE by utilising an ABM approach, which has the ability to provide sufficient insight into QoE and determine how given users behave in a particular situation. The ABM system enables the decision maker to predict and understand the factors behind QoE, and to adjust the value of KPI in a way to improve KQI, thereby increasing the number of Promotors, whilst decreasing that of Detractors.
Method	Iteration 4 provided a method that defines the solution process of building an ABM as well as perceiving QoE. The solution process includes an interaction model presented in Figure 7-5.
Instantiation	Iteration 4 provided a computational instantiation of the model with a NetLogo interface, shown in Figure 7-8 that can be used by practitioners to examine the behaviour of the end-users and gain insight into the effects of QoE on user behaviour.

8.3.2 Contributions to the knowledge base

Generally, in the IS field, the knowledgebase consists of foundations and methodologies derived from past contributions (Hevner et al. 2004). In terms of the foundation, this research is based on knowledge gained from academic research and WIS professional experience to ensure the rigour and credibility of the development process of QoEWA as well as to develop and implement theoretical and technical frameworks for assessing QoE. In terms of the methodologies, for this research different approaches and techniques were adopted to build and evaluate the efficiency of QoEWA.

Value added to foundations

- First, a QoE approach was drawn upon to address objective and subjective problems related to Web Application from a wider information systems perspective, rather than a network perspective as in (Nguyen et al. 2013; ITU-T 2014; Skorin-kapov & Barakovic 2015; Yamauchi et al. 2015). This adds to the knowledge base an effective method for assessing quality of Web Application from a web architecture point of view – considering web objective and subjective factors associated essentially with the web service provider and consumer.

- Second, to overcome the limitations discussed in the literature regarding QoE assessment (e.g. Hobfeld et al. 2012; Zieliński et al. 2012; Nguyen et al. 2013; ITU-T 2014), QoEWA was employed using several models and methods for the development/build phase to construct the artefacts required for assessing QoE. Specifically, the Actual-Vs-Target approach was employed to quantify QoE, ML was utilised to predict it, MOO was deployed to optimise it and ABM was employed to perceive it. Based on the evaluation conducted in each chapter, it was found that these models and their instantiations add new significant dimensions to the knowledge base, in that they provide tailored solutions for the web service community.

- Third, in terms of implementation, this research has involved utilising technical reference models and instantiations (e.g. UML model, Weka knowledge flow model and ABM model) to enable the QoE community to address QoE problems not only from a network and engineering perspective, but also from an IS. Consequently, this allows QoE modellers to consider the IS aspects when they construct QoE models that relate to IS services that are provided over the internet, e.g. web-based applications.

Value added to methodologies

Since QoE was adopted from a multimedia and engineering domain to Web Application, it was necessary to follow a methodology that would assist the web community with the IS bases and guidelines for the 'justify-evaluate' phase. Regarding which, it emerged from the literature that DSR could be adopted to acquire applicable knowledge and techniques provided by prior research. Subsequently, this knowledge base was built upon with newly acquired information collected from experiments and tests based on the QoEWA system. The implementation of QoEWA was performed with an agile development process, with the capability of breaking the development's activities into set-by-step increments with minimal advances planning. This aligns with the DSR methodology as a practical approach (Vidgen et al. 2011) that combines DSR process with the agile process to develop an efficient software system (Aaen 2008). The evaluation of QoEWA was conducted by several models and techniques, including gap analysis and cross-validation to validate and verify the proposed artefacts (e.g. gap analysis was used to validate and verify the quantification and optimisation processes and cross validation was utilised to evaluate the performance of the ML algorithms).

In sum, based on the foundations and methodologies discussed above, it can be concluded that this research adds significant contributions to the knowledge base of both Web Information Systems and QoE research.

8.3.3 Contributions to the environment

The environment defines the problem-solving space, in which the problem space drives the learning, while the solution space includes features and alternatives used to improve a specific application (Hevner et al. 2004; Hevner & Gregor 2013). In this research, the contribution to the environment was recognised when the QoEWA solution was utilised in the context of Web Application within a UK University, whereby continuing issues around service quality, led to a continuous improvement solution to this being sought. It is observed that when QoEWA was utilised, the number of complaints raised by the users against the quality of service was dramatically reduced (from 160 remedy requests to 75). This confirms that QoEWA has the ability to provide the features and tools required for the QoE assessors (e.g. service manager and customer service manager) to identify the problems with the service and as a result improve QoE. In addition, the time needed for support for and maintenance of the Web Application is considerably reduced (i.e. approximately 20% of the support time can be allocated to other activities), which allows the University to optimise resources. In sum, this practice allows the practitioners and the University to share knowledge about the technology adopted to perform such improvement. Building upon this and the evaluation conducted in each iteration, the contributions to the environment can be generally summarised as follows.

- **Contributions to people** (academic and practical audiences): Since QoEWA defines a set of QoE factors derived from web quality requirements and multi-tier architecture, different people with different interests (Gonzalez et al. 2013) can obtain benefits from QoE assessment. At an operational level, the web service manager who looks after web performance and reliability can issue an assessment based on factors (F1, F2, F3, F4, and F5), which are derived from the server-side. The application manager, who oversees usability, can assess QoE based on factors related to usability, such as (F6, F7, F8, and F9). The customer services manager, who is concerned with customer queries and complaints, can assess QoE based on factors related to customer support such as (F10 and F11). At the strategic level, the decision maker, who defines strategy and leads change, can assess QoE using any factor, depending on the proposed plan. From research point of view, this research has created further interest in assessing QoE of Web Application.

- **Contributions to the organisation:** QoEWA has the flexibility and extendibility to be applied to any web service provided over the internet (or customer-supplier environment). It incorporates objective factors into KPIs and subjective factors into KQIs. This enables the organisation to create the assessment based on interfaces (KPIs and KQIs), i.e. add new factors or remove unused ones from the assessment, without making any change to the measurement process. In term of business process, QoEWA is designed with four main assessment processes: quantification, prediction, optimisation, and perception of QoE. These processes together enable the organisation to have an enterprise framework for managing and accessing QoE (Baraković & Skorin-Kapov 2013).

- **Contributions to Technology:** Whilst QoE was originally introduced for multimedia and network services (Kalevi Kilkk 2008), for this research study QoE was adopted to develop a novel model for assessing and evaluating it for Web Application. QoEWA has been designed and implemented to be deployed as a QoE software tool that operates in the back-end to monitor and process the objective data (extracted from CRM) and subjective data (extracted from middleware). Thus, QoEWA can be applied as a technology for assessing QoE or as a part of a CRM system to support customer experience management. Furthermore, it can applied as an architectural framework for managing QoE from an IT perspective, rather than a multimedia and network one.

8.4 Overall Evaluation

In line with the DSR methodology, the practice is to consider the key guidelines of DSR when the developed artefacts are evaluated. In this research, the primary artefacts that have been developed are the QoEWA (conceptual) model and its (technological) instantiation – both forms are considered as legitimate in DSR terms. They are carried out with mindful awareness of the debate on theory within the DSR literature – particularly the anatomy of a design theory (Jones et al. 2007; Walls et al. 1992). For brevity, the work here is presented according the previously published tenets of a design theory, as illustrated in Table 8-5

Table 8-5: Overall evaluation

Component (Jones et al. 2007)	**QoEWA Response**
Purpose and scope	To address the challenge to facilitate and quantify the relationship between the KPIs and KQIs of QoE. Pragmatically, the purpose is to enable service providers to make more informed decisions regarding service delivery and customer satisfaction and/or to optimise resources accordingly.
Constructs	Represented in the core QoEWA model, which initially computes the so-called ‘objective’ and ‘subjective’ for determining KPI and KQI. QoEWA is constructed with four main constructs that quantify, predict, optimise and perceive QoE.
Principle of form and function	Represented in the measures underlying the QoEWA model, the means by which they are aggregated per construct and the means by which the constructs are combined to evidence the QoE assessment. Broadly illustrated in Figure 3-5.
Artefact mutability	The notion of mutability is addressed, in part, in the purposeful exposition of the iterations. Mutability is addressed in more general terms via the separation of measures from constructs: As a design

	principle, the QoEWA can be specialised to different contexts of use via the specialisation of measures and/or the addition (or removal) of constructs (see discussion below).
Statements made are testable propositions	Testable propositions are presented at the micro-level in the testing, results and evaluations of Iterations 1, 2, 3, and 4. At the macro-level the proposition is that the QoEWA will enable service providers to make more informed decisions regarding service delivery and customer satisfaction and/or to optimise resources accordingly.
Justificatory knowledge is provided	Specifically, the underlying knowledge that has informed the design here is in-and-around existing work related to QoE (e.g. Mirkovic et al. 2014) and machine learning techniques associated with improving the understanding of aspects of these (e.g., Mushtaq et al., 2012). It is accepted that this is a more technical than social scientific conception of kernel theory. However, it is one that arguably aligns well with theory being considered as a means by which design knowledge is captured, formalised and communicated. Kernel theory in this sense is the input that provides a basis for aspects of the design.
Principles of implementation	The principles of the implementation are shown primarily in the form of the equations for QoE calculation, standard software development communication techniques, e.g. use cases, class diagrams, etc., machine learning algorithms and an ABM system.
Expository instantiation	Instantiations exist both in the form of the QoEWA model and its computational implementation. The model has both generic and specific forms – the latter populating the metrics observed on the University systems used in this case.

Three points are raised in connection with the table above:

- First, it has been argued that, constructs, models and methods are one type of thing and can be equated to the components of a theory, while instantiations are different (Jones et al. 2007). This is a more pragmatic view of design theory and one this researcher accords with here: The constructs of QoEWA and their relations produce the model (which represents the design theory), whilst the software instantiation is the material artefact that makes the said design theory 'concrete' within its domain of application.

- Second, it is important to specify the degree of mutability of QoEWA as both a model and artefact. Adaptation and/or evolution of the model is feasible in terms of the: (a) Constructs, as KPIs and KQIs may be appended depending on the nature of the domain; (b) measures which, practically, are constrained by the availability of the data via system interfaces; (c) formulaic method(s) by which the QoE is constructed; and (d) the machine learning methods by which MOS is achieved.

- Third, this research has involved considering the testable proposition and notions of generalisation. The proposition at the outset was that QoEWA will enable service providers to make more informed decisions regarding service delivery and customer satisfaction and/or to optimise resources accordingly. It is this proposition that defines the utility of the artefact(s), which, in prototypical form, is most akin to 'new technology X (when applied properly) will provide improvements of Type Y' (Venable 2006 pp 13). The current research in line with this perspective in that existing artefacts that were clearly limited have been modified and hence, improved (Gregor and Hevner 2013). The other key aspect relates to how generalisable the mapping between the problem and solution space is – the degree to which the design knowledge developed and applied in a specific situation can be followed in a similar situation (Prat et al. 2014; Venable 2006; Gregor 2009; Gregor and Hevner 2013). In this regard, this researcher respects the ideographic nature of the design context and goes no further than to assert that, in detailed terms, QoEWA is only an approximation to what might work in other contexts, i.e. this design theory is nascent (Gregor and Hevner 2013). Importantly, however, a strong juxtaposition between generalisation and mutability of the artefact is observed – it is the latter, via the adaptation/evolution mechanisms noted above, that allows for the artefacts to be applied across other domains; the performance of which constitutes future work.

8.5 Meeting the Research Objectives

This section describes how the research objectives have been fulfilled. There were four main objectives formulated at the beginning of this research study (see Section 1.5).

- **Objective 1:** Present a review of the state of the art in QoE of Web Application and associated disciplines to elicit understanding of the challenges related to QoE assessments process: This objective was achieved in Chapter 2, which presented the relevant literature in the fields of Web Application and QoE. The chapter addressed the fundamental concerns of Web Application quality from both the service provider and user perspectives. In addition, the chapter discussed these concerns in the context of QoE, and showed how they can be alleviated.

- **Objective 2:** Conceptualise a solution based on the correlation between the KPIs and KQIs that facilitates the assessment of QoE of Web Application with consideration about the aspects related to web quality requirements and web architecture: This objective was achieved in Chapter 3, which presented the research methodology behind the development of QoEWA. The chapter conceptualised a solution framed by the DSR methodology to provide guidelines on how to develop a theoretical and technical framework for assessing QoE. In addition it briefly described the iterations which build up QoEWA to quantify, predict, optimise, and perceive QoE.

- **Objective 3:** Design and develop a theoretical and technical artefacts that have the capability of mapping the KPIs and KQIs as well as quantifying, predicting, optimising, and perceiving the QoE of Web Application: This objective was achieved by four iterations across four chapters (Chapters 4, 5, 6, and 7). Chapter 4 utilised the Actual-vs-Target approach to quantify QoE, whilst Chapter 5 deployed ML to predict QoE and Chapter 6 employed the MOO approach to optimise QoE. Chapter 7 utilised an ABM approach to perceive and gain insight into QoE. The output of each iteration includes construct, model, method, and an instantiation.

- **Objective 4:** Demonstrate the features of the developed model (QoEWA) and evaluate its capabilities and limitations in the context of Web Application and DSR: This objective was achieved in Chapters 4, 5, 6, and 7. In Chapter 4, two tests were conducted: The first examined the correlation between KPIs and KQIs and the second probed how the model quantifies QoE. In Chapter 5, five tests for five ML algorithms were carried out and DT algorithm was selected as the best. In Chapter 6, the first of two tests examined the correlation between KPIs and KQIs, whilst the second determined the curve of Pareto efficiency to find the best optimal value of QoE. In Chapter 7, two tests were executed to examine user behaviour and interactions. Finally, in Chapter 8 (this chapter) the overall evaluation and contributions have been presented.

8.6 Research Limitations and Future Work

Whilst this research has involved developing a novel QoE framework that draws on the foundations of IS principles to implement a QoE model for Web Application, the assessment of QoE is still a challenging process due to its subjective nature. With reference to the tests and evaluations performed for each developed iteration, the limitations of the research and proposals for future work are discussed next.

8.6.1 QoEWA Limitations

- First, based on the monitoring process conducted in this research to extract the data of the objective metrics, it was noticed that there is a limitation placed by the monitoring tools embedded within the Web Service software to access and extract the objective metrics defined in Table 3-1.

- Second, based on the calculation conducted in Iteration 1, it was noticed that the weight, priority and importance of the subjective factors are subject to change over time, depending on customer requirements, which also evolve over time. Hence, the subjective factors need to be dynamically associated with customer requirements.

- Third, the consistency and the correlation between KPIs and KQIs are not always reliable – the dataset used in QoEWA has a consistent relationship between the two as well as strong positive correlation. However, QoEWA needs to be tested in a scenario where the relationship between the KPIs and KQIs is not consistent and has negative correlation.

- Fourth, the prediction rules are derived from one-off standalone dataset in QoEWA, which prevents having an actionable decision based on dynamic real-time prediction, i.e. the prediction rules are hardcoded in Iteration 3 and Iteration 4.

- Fifth, the price of the technical resources in QoEWA is not considered when QoE is optimised, i.e. in Iteration 3, the resources are optimised based on the service quality and user satisfaction.

- Sixth, a lack of basic socio-demographic characteristics data for examining users' interaction in the ABM system, i.e. the ABM in Iteration 4 considered only location (department) as a demographic factor, due to the data protection aspects.

8.6.2 Future Work

With the above limitations in mind, proposals are made next for refining and improving the proposed model, which could be addressed in future work.

- First, in order to improve the monitoring and measurement process of the objective factors of Web Application, a software architecture needs to be built and integrated with the Web Server, in which the data of QoS metrics are collected independently from the functionality of the Web Service. This will facilitate the data collection of QoS metrics as well as enable the dynamic-ability of measuring QoE. This work could not be achieved without investigating the architecture and the APIs of QoS metrics of the existing Web Services. Eventually, this work will lead to the development of an efficient QoE tool that works independently and dynamically in a real-time environment.

- Second, in order to identify and monitor the subjective factors, a QoE strategy needs to be clearly scoped covering the objective metrics extracted from the server side and the subjective metrics defined in the CRM system (i.e. metrics that are used for customer polling). The strategy will outline in detail the procedure for identifying and formulating the subjective factors based on the business goal and technology adopted. Thus, the subjective factors will be formulated in a way not only reflecting the user's satisfaction, but also indicating the weight, importance and influence of these factors.

- Third, whilst the Actual-Versus-Target approach adopted in this research for quantifying QoE was efficiently performed for the provided scenario and dataset, other approaches are worthy of investigation and so too, different business scenarios, e.g. performing gap analysis between the objective and subjective factors based on values (psychological scale) that are determined by psychologists and/or experts in QoE of Web Application.

- Fourth, with regard to QoE prediction, the ML algorithms applied in this research provided reliable and consistent results with the given dataset.

However they may need to be tested on dynamic datasets and other application domains based on the same factors identified in this research (e.g. [ISO/IEC 2008]). This could help to: (1) take actionable decision; and (2) validate the results and improve the efficiency of the prediction model for a broader set of services and applications and under different scenarios.

- Fifth, the optimisation model in this research focuses on the trade-off between the KPI and KQI values. However, since the optimisation process of QoE is dealing with the network resources, it is important to integrate this process with a cost-price model that delivers a balance between the cost of optimising QoE and the budget allocated for maintaining and operating the provided service. This would make the optimal results more realistic and reasonable.

- Sixth, despite the advantage of incorporating an ABM approach into QoEWA, the ABM system would be more efficient, if other socio-demographic characteristics (such as age, gender, ethnicity, education, occupation, amongst others) were considered and added to the model for examining user behaviour and interaction. Other factors also need to be considered in relation to understanding users' intention and behaviour, such as factors mentioned by Laghari et al. (2011), including: Personal attitude factors (e.g. ease of use, usefulness, complexity, annoyance etc.); social factors (e.g. friend & family, legal issues, organisational pressures etc.); and perceived behavioural control factors (e.g. cost, customer services, etc.).

The future work should involve investigating the trends of cloud technology and virtual applications, where multi-providers are providing different type of services through one player. A question that needs to be answered in this regard is: How can QoE be managed across multiple-players given the challenge of exchanging QoE data between them? QoE prediction will fail if there is a lack of information. Alternatively, public data extracted from social networks could be taken as a suggested solution to providing an information flow channel as inputs for the proposed model.

References

Aaen, I., 2008. Essence: facilitating software innovation. *European Journal of Information Systems*, 17(5), pp.543–553.

Agboma, F. & Liotta, A., 2008. QoE-aware QoS management. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia MoMM 08*. Linz, Austria.: ACM Press, pp. 111–116. Available at: http://portal.acm.org/citation.cfm?doid=1497185.1497210.

Agboma, F. & Liotta, A., 2012. Quality of experience management in mobile content delivery systems. *Telecommunication Systems*, 49(1), pp.85–98. Available at: http://www.springerlink.com/content/9x8p50446q2482r1/.

Akkiraju, R. et al., 1998. Multi machine scheduling: An agent-based approach. In *Proceedings of the 1998 10th Conference on Innovative Applications of Artificial Intelligence, IAAI*. Madison, Wisconsin: American Association of Artificial Intelligence, pp. 1013–1018. Available at: www.aaai.org.

Al-Moayed, A. & Hollunder, B., 2011. An Approach to Model , Configure and Apply QoS Attributes to Web Services. In *The Sixth International Conference on Software Engineering Advances (ICSEA)*. Barcelona, Spain, pp. 405–410.

Al-Moayed, A. & Hollunder, B., 2010. Quality of Service Attributes in Web Services. In *2010 Fifth International Conference on Software Engineering Advances*. Nice, France: IEEE, pp. 367–372. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5614953 [Accessed June 6, 2014].

Alhamazani, K. et al., 2015. Cross-layer multi-cloud real-time application QoS monitoring and benchmarking as-a-Service framework. *IEEE Transactions on Cloud Computing*.

Alreshoodi, M. & Woods, J., 2013. Survey on QoE\ QoS Correlation Models For Multimedia Services. *International Journal of Distributed and Parallel Systems (IJDPS) Vol.4, No.3, May 2013*, 4(3), pp.53–72. Available at: http://arxiv.org/abs/1306.0221 [Accessed January 6, 2015].

Ameigeiras, P. et al., 2010. QoE oriented cross-layer design of a resource allocation algorithm in beyond 3G systems. *Computer Communications*, 33(5), pp.571–582. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0140366409002837 [Accessed October 4, 2013].

Andersson, J. & Andersson, J., 2000. A survey of multiobjective optimization in engineering design. In *Technical report LiTH-IKP-R-1097, Department of Mechanical Engineering, Linköping University*. Linköping, Sweden: Department of Mechanical Engineering, Linkoping University, Linkoping, Sweden, Technical Report No: LiTH-IKP, p. 34.

Aroussi, S. & Mellouk, A., 2014. Survey on machine learning-based QoE-QoS correlation models. In *The 2nd International Conference on Computing, Management and Telecommunications, ComManTel 2014*. Da Nang, Vietnam, pp.

200–204. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6825604 [Accessed January 6, 2015].

ATNET, 2011. *Advanced Technology in Networking*, IRISA (Institute for Research in IT and Random Systems). Available at: https://www.irisa.fr/sites/default/files/atnet_2011.pdf.

Babar, M., 2008. Assessment of a framework for designing and evaluating security sensitive architecture. In *12th Interntaional Conference on Evaluation and Assessment in Software Engineering*. Italy: University of Limerick, pp. 156–165. Available at: http://www.bcs.org/upload/pdf/ewic_ea08_paper17_1.pdf.

Balachandran, A. et al., 2013. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM Computer Communication Review*. Hong Kong, China: ACM, pp. 339–350. Available at: http://dl.acm.org/citation.cfm?doid=2486001.2486025.

Baraković, S., Baraković, J. & Bajrić, H., 2010. QoE Dimensions and QoE Measurement of NGN Services. In *Proceedings of the 18th Telecommunications Forum*. Serbia, Belgrade: TELFOR 2010, pp. 15–18.

Baraković, S. & Skorin-Kapov, L., 2013. Survey and Challenges of QoE Management Issues in Wireless Networks. *Journal of Computer Networks and Communications*, 2013, pp.1–28. Available at: http://www.hindawi.com/journals/jcnc/2013/165146/.

Basili, V.R., Caldiera, G. & Rombach, H.D., 1994. The goal question metric approach. In *Encyclopaedia of Software Engineering*. Wiley, pp. 528–532. Available at: http://maisqual.squoring.com/wiki/index.php/The Goal Question Metric Approach.

Bass, L., Klein, M. & Bachmann, F., 2002. Quality attribute design primitives and the attribute driven design method. In *4th International Workshop, PFE 2001*. Berlin, Heidelberg: Springer, pp. 169–186. Available at: http://link.springer.com/chapter/10.1007/3-540-47833-7_17.

Batteram, H. et al., 2014. Delivering Quality of Experience in Multimedia Networks. *Bell Labs Technical Journal*, 18(4), pp.3–17.

Behkamal, B., Kahani, M. & Akbari, M.K., 2009. Customizing ISO 9126 quality model for evaluation of B2B applications. *Information and Software Technology*, 51(3), pp.599–609. Available at: http://dx.doi.org/10.1016/j.infsof.2008.08.001.

Benveniste, A. et al., 2008. Probabilistic QoS and Soft Contracts for Transaction-Based Web Services Orchestrations. *IEEE Transactions on Services Computing*, 1(4), pp.187–200. Available at: http://www.labri.fr/perso/anca/docflow/publications.html.

Bevan, N., Carter, J. & Harker, S., 2015. ISO 9241-11 Revised: What Have We Learnt About Usability Since 1998? In *International Conference on Human-Computer Interaction*. Switzerland: Springer International Publishing., pp. 143–151. Available at: http://link.springer.com/10.1007/978-3-319-20901-2.

Bonabeau, E., 2002. Agent-based modeling: methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(3), pp.7280–7287. Available at: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=128598&tool=pmcentre

z&rendertype=abstract.

Bouten, N. et al., 2013. A Multicast-Enabled Delivery Framework for QoE Assurance of Over-The-Top Services in Multimedia Access Networks. *Journal of Network and Systems Management*, 21(4), pp.677–706. Available at: http://link.springer.com/10.1007/s10922-013-9262-8.

Brooks, P. & Hestnes, B., 2010. User measures of quality of experience: Why being objective and quantitative is important. *IEEE Network*, 24(2), pp.8–13.

Calero, C., Ruiz, J. & Piattini, M., 2004. A web metrics survey using WQM. In *ICWE 2004*. Berlin Heidelberg: Springer-Verlag, pp. 147–160. Available at: http://link.springer.com/chapter/10.1007/978-3-540-27834-4_19.

Calero, C., Ruiz, J. & Piattini, M., 2005. Classifying web metrics using the web quality model. *Online Information Review*, 29(3), pp.227–248.

Cassidy, L. & Hamilton, J., 2016. A design science research approach to website benchmarking. *Benchmarking: An International Journal*, 23(5), pp.1054–1075.

Cecchet, E. et al., 2013. mBenchLab: Measuring QoE of Web applications using mobile devices. In *IEEE/ACM International Symposium on QoS (IWQoS2013)*. Montreal, Canada, pp. 1–10.

Chang, Y.C. et al., 2010. Online game QoE evaluation using paired comparisons. In *2010 IEEE International Workshop Technical Committee on Communications Quality and Reliability, CQR 2010*. Vancouver, BC, Canada: IEEE, pp. 1–6.

Chieu, T.C. et al., 2009. Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment. In *2009 IEEE International Conference on e-Business Engineering*. Macau, China: IEEE, pp. 281–286. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5342101.

COMBO, 2013. *Monitoring parameters relation to QoS / QoE and KPIs*, Spain: COMBO. Available at: http://ict-combo.eu/data/uploads/pdf-combo-v2/combo_d4.1_wp4_20june2014_aitia_v2.0_sec.pdf.

Conallen, J., 1999. Modeling Web application architectures with UML. *Magazine Communications of the ACM*, 42(10), pp.63–70. Available at: http://dl.acm.org/citation.cfm?id=317677.

Das, M.S., Govardhan, A. & Lakshmi, D., 2016. Best practices for web applications to improve performance of QoS. In *ICTCS '16 Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. Udaipur, India: ACM, p. 123.

Deb, K., 2011. Multi-Objective Optimization Using Evolutionary Algorithms : An Introduction. *KanGAL Report, (2011003)*, pp.1–24.

Dillon, E. et al., 2009. PERIMETER: A Quality of Experience Framework. In *International Workshop on the Future Internet of Things and Services*. Berlin, Germany. Available at: http://repository.wit.ie/1496/.

Dillon, E., Power, G. & Grant, F.C., 2010. Experimental Testing in the Future Internet PERIMETER Project. In G. Berlin, ed. *Future Internet - FIS 2010 - Proceedings of the Third Future Internet Symposium, Berlin, Germany, September 20-22, 2010.*

Springer, pp. 30–39. Available at: http://link.springer.com/chapter/10.1007/978-3-642-15877-3_4.

Dorst, K. & Cross, N., 2001. Creativity in the design process, convolution of problem-solution. *Design studies*, 22(5), pp.425–437.

Drogseth, D., 2009. *EMA TM Advisory Note : An Adopter ' s Guide to User Experience Management – How to Pick the Right QoE Solution for You*, Enterprise Management Associates, Inc.

Dubé, L. & Paré, G., 2003. Rigor in Information Systems Positivist Case Research: Current Practices, Trends, and Recommendations. *MIS Quarterly*, 27(4), pp.597–635. Available at: http://www.jstor.org/stable/30036550.

Dusi, M. et al., 2012. A closer look at thin-client connections: Statistical application identification for QoE detection. *IEEE Communications Magazine*, 50(11), pp.195–202.

Eckerson, B.W.W., 2009. Performance Management Strategies: How to Create and Deploy Effective Metrics. *The Data Warehousing Institute TDWI*, pp.1–32.

Elkotob, M. et al., 2010. Multimedia QoE optimized management using prediction and statistical learning. In *35th Annual IEEE Conference on Local Computer Networks*. Denver, Colorado, USA: Ieee, pp. 324–327. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5735733.

Fiedler, M., Hossfeld, T. & Tran-Gia, P., 2010. A generic quantitative relationship between quality of experience and quality of service. *IEEE Network*, 24(2), pp.36–41. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5430142.

Finkelstein, A.C.W. et al., 2002. Ubiquitous Web Application Development - A Framework for Understanding. In *Proc. 6th World MultiConf. on Systemics, Cybernetics and Informatics (SCI 2002)*. Orlando, Florida, USA, pp. 431–438.

Fleming, P.J. et al., 2005. Many-Objective Optimization: An Engineering Design Perspective. In *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, Berlin, Heidelberg, pp. 14–32.

Franca, J. & Soares, M.S., 2015. SOAQM : Quality model for SOA applications based on ISO 25010. *ICEIS*, (2), pp.60–70.

Gardlo, B. et al., 2011. A QoE Evaluation Methodology for HD Video Streaming Using Social Networking. In *IEEE International Symposium on Multimedia (ISM 2011)*. Dana Point, California, USA: Ieee, pp. 222–227. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6123350 [Accessed November 11, 2013].

Gay, B. & Weaver, S., 2011. Theory building and paradigms: A primer on the nuances of theory construction. *American International Journal of Contemporary Research*, 1(2), pp.24–32. Available at: http://www.aijcrnet.com/.

Geerts, D., Moor, K. De & Ketyko, I., 2010. Linking an integrated framework with appropriate methods for measuring QoE. In *Quality of Multimedia Experience (QoMEX), 2010 Second International Workshop*. Trondheim, Norway: IEEE, pp.

158–163. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5516292 [Accessed August 20, 2013].

Gilbert, N. & Terna, P., 2000. How to build and use agent-based models in social science. *Mind & Society*, 1(1), pp.57–72. Available at: http://dx.doi.org/10.1007/BF02512229.

Gómez, G. et al., 2013. Towards a QoE-Driven Resource Control in LTE and LTE-A Networks. *Journal of Computer Networks and Communications*, 2013, pp.1–15. Available at: http://www.hindawi.com/journals/jcnc/2013/505910/.

Gonzalez, A.J. et al., 2013. New Framework and Mechanisms of Context-Aware Service Composition in the Future Internet. *ETRI Journal*, 35(1), pp.7–17. Available at: http://etrij.etri.re.kr/etrij/journal/article/article.do?volume=35&page=7.

Gregor, S., 2009. Building theory in the sciences of the artificial. In *09 Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*. Philadelphia, Pennsylvania: ACM, p. 10. Available at: http://portal.acm.org/citation.cfm?doid=1555619.1555625.

Grimm, V. et al., 2010. The ODD protocol: A review and first update. *Ecological modelling*, 221(23), pp.2760–2768. Available at: http://dx.doi.org/10.1016/j.ecolmodel.2010.08.019.

Guse, D. et al., 2014. Web-QoE under real-world distractions: Two test cases. In *Quality of Multimedia Experience (QoMEX), 2014 Sixth International Workshop*. Singapore: IEEE, pp. 220–225.

Guyard, F. & Beker, S., 2010. Towards real-time anomalies monitoring for QoE indicators. *Annales des Telecommunications/Annals of Telecommunications*, 65(1–2), pp.59–71.

Habachi, O. et al., 2012. MOS-Based Congestion Control for Conversational Services in Wireless Environments. *IEEE Journal on Selected Areas in Communications*, 30(7), pp.1225–1236. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6248265&contentType=Journals+%26+Magazines&matchBoolean%3Dtrue%26rowsPerPage%3D30%26searchField%3DSearch_All%26queryText%3D%28p_Title%3A%22MOS-based+Congestion+Control+for+Conversational+Servi.

Hasan, M., Jaafar, J. & Hassan, M., 2012. Monitoring web services' quality of service: a literature review. *Artificial Intelligence Review*, 42(4), pp.835–850. Available at: http://dx.doi.org/10.1007/s10462-012-9358-7.

Hassouna, M. & Arzoky, M., 2011. Agent based modelling and simulation: Toward a new model of customer retention in the mobile market. *Proc. of the 2011 Summer Computer Simulation Conference*, (March), pp.30–35. Available at: http://dl.acm.org/citation.cfm?id=2348201.

Hevner, A. et al., 2004. Design science in information systems research. *MIS quarterly*, 28(1), pp.75–105. Available at: http://www.springerlink.com/index/pdf/10.1007/s11576-006-0028-8 [Accessed November 28, 2014].

Hevner, A. & Chatterjee, S., 2010. Design Research in Information Systems. In *The series Integrated Series in Information Systems*. Integrated Series in Information Systems. Boston, MA: Springer US, pp. 9–22. Available at: http://link.springer.com/10.1007/978-1-4419-5653-8 [Accessed July 9, 2014].

Hevner, A. & March, S., 2003. IT systems perspectives - the information systems research cycle. *Computer-IEEE Computer Magazine*, 36(11), pp.111–113. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1244541 [Accessed January 7, 2015].

Hevner, A.R. & Gregor, S., 2013. Positioning and Presenting Design Science Research for Maximum Impact. *MIS quarterly*, 37(2), pp.1–6.

Hobfeld, T. et al., 2012. Challenges of QoE Management for Cloud Applications. *IEEE Communications Magazine*, 50(4), pp.58–65.

Hoßfeld, T., Tran-Gia, P. & Fiedler, M., 2007. Quantification of Quality of Experience for Edge-Based Applications. *Managing Traffic Performance in Converged Networks*, 4516, pp.361–373. Available at: http://dx.doi.org/10.1007/978-3-540-72990-7_34.

Hsu, W.H. et al., 2013. Web-based QoE measurement framework. In *Signal-Image Technology & Internet-Based Systems (SITIS), 2013 International Conference*. Kyoto, Japan: IEEE, pp. 265–272.

Hummel, A. et al., 2012. An Agent-Based Simulation of Viral Marketing Effects in Social Networks. In *26th European Simulation and Modelling Conference*. Essen, Germany, pp. 212–219.

Hussain, A. & Kutar, M., 2009. Usability metric framework for mobile phone application. In *The 10th Annual PostGraduate Symposium on The Convergence of Telecommunications, Networking and Broadcasting*. Liverpool, UK: PGNet, pp. 1–5. Available at: http://www.cms.livjm.ac.uk/pgnet2009/Proceedings/Papers/2009013.pdf [Accessed June 5, 2014].

IBM, 2011. Smart SOA Solutions with WebSphere Enterprise Service Bus Registry. *IBM WebSphere*.

ISO-42010, 2012. Systems and software engineering - Architecture description. *IEEE*, 2011.

ISO/IEC, 2008. ISO/IEC JTC1/SC7 N4098. *Software and Systems Engineering Secretariat: CANADA (SCC)*, (V. 0.55), pp.1–42.

ISO/IEC TR 9126-3, 2002. Software engineering – Product quality – Part 3: Internal metrics. *Software Engineering Secretariat: CANADA (SCC)*. Available at: ISO.

ISO 9241-11, 1998. ISO 9241-11:1998, Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability. , p.28. Available at: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en.

ITU-T, 2005. Recommendation G.1030: Estimating End-To-End Performance in IP Networks for Data Applications. *ITU-T G-Series, The International Telecommunication Union (ITU)*.

ITU-T, 2014. Recommendation G.1031: QoE factors in web-browsing. *ITU-T G-Series, The International Telecommunication Union (ITU)*.

ITU-T, 2006a. Recommendation P.10/G.100: Vocabulary for performance and quality of service. *ITU-T P-Series, The International Telecommunication Union (ITU)*.

ITU-T, 2006b. Recommendation P. 800.1: Mean opinion score (MOS) terminology. *ITU-T P-Series, The International Telecommunication Union (ITU)*.

Ivesic, K., Matijasevic, M. & Skorin-kapov, L., 2011. Simulation Based Evaluation of Dynamic Resource Allocation for Adaptive Multimedia Services. In *The 7th International Conference on Network and Services Management. (CNSM)*. Paris, France, pp. 432–435.

Ivesic, K., Skorin-Kapov, L. & Matijasevic, M., 2014. Cross-layer QoE-driven admission control and resource allocation for adaptive multimedia services in LTE. *Journal of Network and Computer Applications*, 46, pp.336–351. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1084804514002173.

Janssen, M., 2005. Integrating Optimisation and Agent-based Modelling. In *Proceedings 28th European Conference on Modelling and Simulation (ECMSs)*. Brescia, Italy, pp. 775–781. Available at: http://isecoeco.org/pdf/agent_based _modeling.pdf.

Jones, D. et al., 2007. The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5), pp.312–335. Available at: http://aisel.aisnet.org/jais/vol8/iss5/1/ [Accessed December 16, 2014].

Kalevi Kilkk, 2008. Quality of Experience in Communications Ecosystem. *Journal Of Universal Computer Science*, 14(5), pp.615–624. Available at: http://www.jucs.org/jucs_14_5/quality_of_experience_in.

Kan, S.H., Parrish, J. & Manlove, D., 2001. In-process Metrics for Software Testing. *IBM Syst. J.*, 40(1), pp.220–241. Available at: http://dx.doi.org/10.1147/sj.401.0220.

Khan, A., Sun, L. & Ifeachor, E., 2012. QoE Prediction Model and its Application in Video Quality Adaptation Over UMTS Networks. *IEEE Transactions on Multimedia*, 14(2), pp.431–442. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6082445.

Kim, H.J., Lee, K.H. & Zhang, J., 2010. In-service feedback QoE framework. In *2010 Third International Conference on Communication Theory, Reliability, and Quality of Service*. Glyfada, Greece: IEEE, pp. 135–138.

Konak, A., Coit, D.W. & Smith, A.E., 2006. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, 91(9), pp.992–1007.

De Koning, T. et al., 2007. Of MOS and men: bridging the gap between objective and subjective quality measurements in mobile TV. In *Proc. of SPIE-IS&T Electronic Imaging*. San Jose, CA, USA: SPIE Digital Library, pp. 1–11. Available at: http://discovery.ucl.ac.uk/173401/.

Kotsiantis, S.B., 2007. Supervised machine learning: A review of classification techniques. *Informatica*, 31, pp.249–268. Available at: http://books.google.com/books?hl=en&lr=&id=vLiTXDHr_sYC&oi=fnd&pg=PA3

&dq=survey+machine+learning&ots=CVsyuwYHjo&sig=A6wYWvywU8XTc7Dzp8ZdKJaW7rc%5Cnpapers://5e3e5e59-48a2-47c1-b6b1-a778137d3ec1/Paper/p800%5Cnhttp://www.informatica.si/PDF/31-3/11_Kotsiantis - S.

Krasner, G.E. & Pope, S.T., 1988. A Cookbook for Using the Model- View-Controller User Interface Paradigm in Smalltalk-80. *Journal Of Object Oriented Programming*, 1, pp.26–49.

Kuechler, B. & Vaishnavi, V., 2008. On theory development in design science research: anatomy of a research project. *European Journal of Information Systems*, 17(5), pp.489–504.

Kulik, I. & Trinh, T., 2011. Investigation of quality of experience for 3D streams in gigabit passive optical network. *Energy-Aware Communications*, pp.157–168. Available at: http://www.springerlink.com/index/V3540215X355U730.pdf.

Kumar, N. & Dadhich, R., 2015. Quality Models for Web-based Application : A Comparative Study. *International Journal of Computer Applications*, 125(2), pp.25–32.

Laghari, K. & Connelly, K., 2012. Toward Total Quality of Experience : A QoE Model in a Communication Ecosystem. *Communications Magazine, IEEE*, 50(4), pp.58–65. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6178834 [Accessed June 13, 2013].

Laghari, K.U.R. et al., 2011. QoE aware service delivery in distributed environment. *Proceedings - 25th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2011*, pp.837–842.

LeRouge, C. & Lisetti, C., 2006. Triangulating design science, behavioural science and practice for technological advancement in tele-home health. *International journal of healthcare technology and Management*, 7(5), pp.348–363. Available at: http://inderscience.metapress.com/index/bk3jw7lvw7la0w50.pdf [Accessed December 20, 2014].

Lew, P. et al., 2012. An integrated strategy to systematically understand and manage quality in use for web applications. *Requirements Engineering*, 17(4), pp.299–330. Available at: http://link.springer.com/10.1007/s00766-011-0128-x.

Lilburne, B. & Khan, K., 2004. Measuring Quality Metrics for Web Applications. In *2004 Information Resources Management Association International Conference*. New Orleans, USA, pp. 1–9.

Lima, S.R. & Carvalho, P., 2011. Enabling self-adaptive QoE/QoS control. In *36th Annual IEEE Conference on Local Computer Networks*. Bonn, Germany: IEEE, pp. 239–242.

Lin, A.C., 1998. Bridging Positivist and Interpretivist Approaches to Qualitative Methods. *Policy Studies Journal*, 26(1), pp.162–180. Available at: http://doi.wiley.com/10.1111/j.1541-0072.1998.tb01931.x.

Liotou, E. et al., 2015. Quality of Experience Management in Mobile Cellular Networks: Key Issues and Design Challenges. *Communications Magazine, IEEE*, 53(7), pp.145–153.

Losavio, F. et al., 2003. Quality characteristics for software architecture. *Journal of Object Technology*, 2(2), pp.133–150. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-0347973763&partnerID=40&md5=d801656b979c2cf6a3aeb5dfb7f4b35a.

Macal, C.M. & North, M.J., 2007. Agent-based modeling and simulation: Desktop ABMS. In *Proceedings - Winter Simulation Conference*. IEEE, pp. 95–106.

Macal, C.M. & North, M.J., 2005. Tutorial on agent-based modeling and simulation. In *Proceedings of the 37th conference on Winter simulation*. ACM, pp. 2–15.

Madhavan, R. & Alagarsamy, K., 2014. Usability Evaluation of Object Oriented Applications Using Learning Curves. *International Journal of Engineering and Technical Research (IJETR)*, 2(2 ISSN: 2321-0869), pp.35–39.

Maes, P., 1991. The agent network architecture (ANA). *ACM SIGART Bulletin*, 2(4), pp.115–120.

Malak, G. et al., 2004. Towards a Multidimensional Model for Web-Based Applications Quality Assessment. In *International Conference on Electronic Commerce and Web Technologies*. Springer, Berlin, Heidelberg, pp. 316–327. Available at: https://webvpn.zih.tu-dresden.de/+CSCO+00756767633A2F2F6A6A6A2E666365766174726579766178 2E70627A++/content/9f8pxcw9vaq2n6x1/.

March, S.T. & Smith, G.F., 1995. Design and natural science research on information technology. *Decision Support Systems*, 15(4), pp.251–266. Available at: http://linkinghub.elsevier.com/retrieve/pii/0167923694000412.

Marez, L. De & Moor, K. De, 2007. The Challenge Of User- And QoE-Centric Research And Product Development In Today ' s ICT-Environment. *Observatorio (OBS) Journal*, 3, pp.1–22.

Marler, R.T. & Arora, J.S., 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6), pp.369–395.

Martinez, L., 2014. QoE : A market perspective analysis. In *25th European Regional Conference of the International Telecommunications Society (ITS)*. Brussels, Belgium, pp. 22–25.

Menkovski, V., Exarchakos, G., Liotta, A. & Sanchez, A.C., 2010. Estimations and Remedies for Quality of Experience in Multimedia Streaming. In *The 2010 Third International Conference on Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services*. Nice, France: IEEE Computer Society, pp. 11–15. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5600357 [Accessed August 19, 2013].

Menkovski, V., Exarchakos, G., Liotta, A. & Sánchez, A.C., 2010. Measuring Quality of Experience on a Commercial Mobile TV Platform. In *2010 Second International Conferences on Advances in Multimedia*. Athens, Greece: Ieee, pp. 33–38. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5501601 [Accessed August 18, 2013].

Menkovski, V., Oredope, A., et al., 2009. Optimized online learning for qoe prediction. In *The 21st Benelux Conference on Artificial Intelligence (BNAIC 2009)*. Belgium, Netherlands, pp. 169–176.

Menkovski, V., Liotta, A., et al., 2009. Predicting Quality of Experience in Multimedia Streaming. In *International Conference on Advances in Mobile Computing and Multimedia*. Kuala Lumpur, Malaysia, pp. 52–59.

Menkovski, V., Exarchakos, G. & Liotta, A., 2010. Online QoE prediction. In *2010 Second International Workshop on Quality of Multimedia Experience (QoMEX)*. Trondheim, Norway: IEEE, pp. 118–123. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5517692.

Mich, L., Franch, M. & Gaio, L., 2003. Evaluating and designing web site quality. *IEEE Multimedia*, 10(1), pp.34–43.

Mifsud, J., 2011. The Difference (and Relationship) Between Usability and User Experience. *Usability Geek*. Available at: http://usabilitygeek.com/the-difference-between-usability-and-user-experience/.

Mifsud, J., 2015. Usability Metrics - A Guide To Quantify The Usability Of Any System. *Usability Geek*. Available at: http://usabilitygeek.com/usability-metrics-a-guide-to-quantify-system-usability/.

Mirkovic, M. et al., 2014. Evaluating the role of content in subjective video quality assessment. *The Scientific World Journal*, 2014, pp.1–9.

Mitra, K., Zaslavsky, A. & Ahlund, C., 2011. Dynamic Bayesian networks for sequential quality of experience modelling and measurement. In *Smart Spaces and Next Generation Wired/Wireless Networking*. Berlin Heidelberg: Springer Berlin, pp. 135–146.

Moller, S. et al., 2009. A taxonomy of quality of service and Quality of Experience of multimodal human-machine interaction. In *2009 International Workshop on Quality of Multimedia Experience*. IEEE, pp. 7–12. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5246986.

De Moor, K. et al., 2010. Proposed Framework for Evaluating Quality of Experience in a Mobile, Testbed-oriented Living Lab Setting. *Mobile Networks & Applications*, 15(3), pp.378–391.

Morais, A. & Cavalli, A., 2012. Managing Customer Experience through Service Quality Monitoring. In *Conference Proceedings of Future Network and Mobile Summit.* Berlin, Germany: IEEE, pp. 1–9.

Mushtaq, M.S., Augustin, B. & Mellouk, A., 2012. Empirical study based on machine learning approach to assess the QoS/QoE correlation. In *17th European Conference on Networks and Optical Communications*. Vilanova i la Geltrú, Spain: IEEE, pp. 1–7. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6249939.

Negash, S., Ryan, T. & Igbaria, M., 2003. Quality and effectiveness in Web-based customer support systems. *Information & Management*, 40(8), pp.757–768. Available at: http://www.sciencedirect.com/science/article/pii/S0378720602001015.

Ngatchou, P., Zarei, a. & El-Sharkawi, a., 2005. Pareto Multi Objective Optimization. *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, pp.84–91. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1599245.

Nguyen, L.T., Harris, R. & Punchihewa, A., 2013. Assessment of quality of experience for web browsing-as function of quality of service and content factors. In IEEE, ed. *International Conference on Ubiquitous and Future Networks, ICUFN*. Danang, Vietnam, pp. 764–769.

Nilsson, N.J., 1997. *Introduction to Machine Learning* Second edi., Massachusetts Institute of Technology (MIT Press). Available at: https://books.google.co.uk/books?id=4j9GAQAAIAAJ&q=isbn:9780262012430&dq=isbn:9780262012430&hl=en&sa=X&ved=0ahUKEwjqzrLtuYbSAhVOGsAKHWQMANcQ6AEIHDAA.

Nokia, 2004. Quality of Experience (QoE) of mobile services : Can it be measured and improved ? , (Nokia code: 11212 – 1004 Indivisual/Libris Copyright © 2004 Nokia.). Available at: www.nokia.com.

O’Brien, L., Merson, P. & Bass, L., 2007. Quality Attributes for Service-Oriented Architectures. In *Systems Development in SOA Environments, 2007. SDSOA ’07: ICSE Workshops 2007. International Workshop*. IEEE.

OASIS, 2006. Reference Model TC." OASIS Reference Model for Service Oriented Architecture 1. *OASIS Open 2006*, pp.1–31. Available at: https://www.oasis-open.org/committees/download.php/15427/wd-soa-rm-10.pdf.

OASIS, 2012. Web Services Quality Factors Version 1 . 0 Candidate OASIS Standard 01. *OASIS Open 2012*, pp.1–29.

Obafemi, O., Gyires, T. & Tang, Y., 2011. An Analytic and Experimental Study on the Impact of Jitter Playout Buffer on the E-model in VoIP Quality Measurement. In *The Tenth International Conference on Networks- ICN2011*. The Netherlands Antilles, pp. 151–156.

Offutt, J., 2002a. Quality attributes of Web software applications. *IEEE Software*, 19(2), pp.25–32.

Offutt, J., 2002b. Web Software Applications Quality Attributes. In *Quality Engineering in Software Technology (CONQUEST 2002)*,. Nuremberg, Germany: IEEE, pp. 187–198. Available at: http://www.cs.gmu.edu/~offutt/rsrch/papers/conquest02.pdf [Accessed June 6, 2014].

Olsina, L. & Rossi, G., 2002. Measuring Web application quality with WebQEM. *IEEE Multimedia*, 9, pp.20–29.

Orlikowski, W.J. & Baroudi, J.J., 1991. Studying Information Technology in Organizations: Research Approaches and Assumptions. *Information Systems Research*, 2(1), pp.1–28. Available at: http://pubsonline.informs.org/doi/abs/10.1287/isre.2.1.1.

Oztekin, A., Nikov, A. & Zaim, S., 2009. UWIS: An assessment methodology for usability of web-based information systems. *Journal of Systems and Software*, 82(12), pp.2038–2050. Available at:

http://linkinghub.elsevier.com/retrieve/pii/S0164121209001642 [Accessed November 20, 2014].

Peffers, K. et al., 2008. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), pp.45–77.

Perkis, A., Munkeby, S. & Hillestad, O., 2006. A model for measuring Quality of Experience. *Proceedings of the 7th Nordic Signal Processing Symposium - NORSIG 2006*, pp.198–201. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4052204.

Perrey, R. & Lycett, M., 2003. Service-oriented architecture. In *Applications and the Internet Workshops - SAINT'03*. Orlando, United States: Orlando, United States, pp. 116–119. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1210138.

Phillips, S.C. et al., 2015. Linking Quality of Service and Experience in Distributed Multimedia Systems Using PROV Semantics. *2015 IEEE Symposium on Service-Oriented System Engineering*, pp.117–126. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7133520.

Prat, N., Comyn-Wattiau, I. & Akoka, J., 2014. Artifact Evaluation in Information Systems Design Science Research-A Holistic View. In *18th Pacific Asia Conference*. Chengdu, China, pp. 1–17. Available at: http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1368&context=pacis2014.

Puliafito, A., Tomarchio, O. & de Meer, H., 1997. An Agent-based Framework for QoS Management. *First World Congress on System Simulation WCSS'97: 4th ANMT on QoS*, pp.392–396.

Qadir, Q.M., Kist, A.A. & Zhang, Z., 2015. Mechanisms for QoE optimisation of Video Traffic : A review paper. *Australasian Journal of Information, Communication Technology and Applications*, 1(1), pp.1–18.

Qadir, S., Kist, A. a & Zhang, Z., 2014. QoE-Aware Cross-Layer Architecture for Video Traffic over Internet. In *Region 10 Symposium, 2014 IEEE*. Kuala Lumpur, Malaysia: IEEE, pp. 522–526.

Qianqian Xu et al., 2010. Bridging the gap between objective score and subjective preference in video quality assessment. In *Multimedia and Expo (ICME), 2010 IEEE International Conference*. Singapore: IEEE, pp. 908–913.

Ramler, R. et al., 2002. A quality-driven approach to web testing. *Ibero-american Conference on Web Engineering, ICWE 2002*, 1, pp.81–95.

Ran, S., 2003. A model for web services discovery with QoS. *ACM Sigecom exchanges*, 4(1), pp.1–10. Available at: http://portal.acm.org/citation.cfm?id=844360.

Rand, W. & Rust, R.T., 2011. Agent-based modeling in marketing: Guidelines for rigor. *International Journal of Research in Marketing*, 28(3), pp.181–193. Available at: http://dx.doi.org/10.1016/j.ijresmar.2011.04.002.

Rawsthorne, P., 2011. MVC in a three-tier architecture. *MVC pattern*. Available at: http://criticaltechnology.blogspot.co.uk/2011/09/mvc-in-three-tier-architecture.html.

Reichheld, F.F., 1996. Learning from customer defections. *Harvard Business Review*, 74(2), pp.56–69.

Reichheld, F.F., 2003. The One Number You Need to Grow. *Harvard Business Review*, 81(12), pp.46–55.

Reichl, P., Tuffin, B. & Schatz, R., 2013. Logarithmic laws in service quality perception: Where microeconomics meets psychophysics and quality of experience. *Telecommunication Systems*, 52(2), pp.587–600.

Rivera, D. et al., 2015. QoE Evaluation Based on QoS and QoBiz Parameters Applied to an OTT Service. In *2015 IEEE International Conference on Web Services*. Francisco, USA: IEEE, pp. 607–614. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7195621.

Ruiz, J., Calero, C. & Piattini, M., 2003. A three dimensional web quality model. *International Conference on Web Engineering , Spr inger*, LNCS 2722, pp.384–385. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-27544513291&partnerID=40&md5=a800301f4b78095e4782d57ccce932cc.

Sampath, S. et al., 2007. Applying concept analysis to user-session-based testing of web applications. *IEEE Transactions on Software Engineering*, 33(10), pp.643–658.

Samuel, A.L., 1959. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, 3(3), pp.210–229.

Sargent, R.G., 2014. Proceedings of the 2014 Winter Simulation Conference A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds. In *Proceedings of the 2014 Winter Simulation Conference*. Savannah, GA: IEEE, pp. 118–131.

Schalles, C., 2013. *Usability evaluation of modeling languages* 1st ed., Springer, Gabler Verlag. Available at: http://link.springer.com/content/pdf/10.1007/978-3-658-00051-6.pdf.

Schatz, R. et al., 2013. From Packets to People: Quality of Experience as a New Measurement Challenge. In *Data Traffic Monitoring and Analysis, Springer*. Verlag Berlin Heidelberg: Springer-Verlag, pp. 219–263. Available at: http://link.springer.com/chapter/10.1007/978-3-642-36784-7_10 [Accessed October 29, 2013].

Schatz, R., Egger, S. & Platzer, A., 2011. Poor, good enough or even better? Bridging the gap between acceptability and QoE of mobile broadband data services. In *IEEE International Conference on Communications ICC2011*. Kyoto, Japan, pp. 1–6.

Schumacher, J. et al., 2010. Providing an User Centric Always Best Connection. In *2nd International Conference on Evolving Internet*. Valencia, Spain, pp. 80–85. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5616515.

Seffah, A. et al., 2006. Usability measurement and metrics: A consolidated model. *Software Quality Journal*, 14(2), pp.159–178. Available at: http://link.springer.com/article/10.1007/s11219-006-7600-8.

Seffah, A., Kececi, N. & Donyaee, M., 2001. QUIM: a framework for quantifying usability metrics in software quality models. *Proceedings Second Asia-Pacific*

Conference on Quality Software, (1), pp.311–318. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=990036.

Seppanen, J. & Varela, M., 2013. QoE-driven network management for real-time over-the-top multimedia services. In *IEEE Wireless Communications and Networking Conference, WCNC*. Shanghai, China: IEEE, pp. 1621–1626.

Sharma, S., Rangaiah, G.P. & Cheah, K.S., 2012. Multi-objective optimization using MS Excel with an application to design of a falling-film evaporator system. *Food and Bioproducts Processing*, 90(2), pp.123–134. Available at: http://dx.doi.org/10.1016/j.fbp.2011.02.005.

Signore, O., 2005. A comprehensive model for Web sites quality. *Seventh IEEE International Symposium on Web Site Evolution*, 2005, pp.30–38. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1517978.

Skorin-kapov, L., 2012. A Multi-Dimensional View of QoE : The ARCU Model. In *The 35th Jubilee International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO '12)*. Opatija, Croatia, pp. 662–666. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6240728 [Accessed April 25, 2014].

Skorin-kapov, L. & Barakovic, S., 2015. Multidimensional modelling of quality of experience for mobile Web browsing. *Computers in Human Behavior*, 50, pp.314–332.

Soldani, D., Li, M. & Cuny, R., 2006. *QoS and QoE Management in UMTS Cellular Systems* D. Soldani, M. Li, & R. Cuny, eds., Chichester, UK: John Wiley. Available at: http://doi.wiley.com/10.1002/9780470034057.

Song, W., Tjondronegoro, D. & Docherty, M., 2012. *Understanding User Experience of Mobile Video : Framework , Measurement , and Optimization*, INTECH Open Access Publisher. Available at: http://eprints.qut.edu.au/48502/.

Staehle, B. et al., 2011. AquareYoum: Application and Quality of Experience-Aware Resource Management for YouTube in Wireless Mesh Networks. *PIK-praxis der Informationsverarbeitung und Kommunikation*, 34(4), pp.144–148.

Stojanovic, M.D., Vukasinovic, M.M. & Djogatovic, V.M.R., 2015. Approaches to Quality of Experience Management in the Future Internet. *Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS), IEEE*, pp.281–288.

Strandberg, C., Wahlberg, O. & Öhman, P., 2011. The influence of active and passive customer behavior on switching in customer relationships. *Managing Service Quality: An International Journal*, 21(5), pp.448–464.

Susila & Vadivel, S., 2014. QoS Measurement Tool for Web Service Selection. *International Journal of Web Engineering*, 3(1), pp.1–8. Available at: http://article.sapub.org/10.5923.j.web.20140301.01.html#Sec1.

Takahashi, A., 2009. Framework and standardization of quality of experience (QoE) design and management for audiovisual communication services. *NTT Technical Review*, 7(4), pp.1–5.

Temnenco, V., Architect, S. & Systems, S., 2010. Using a combined SOA and TOGAF environment for increased productivity : Part 2 . Service-oriented architecture from top to bottom. *IBM - DeveloperWorks*, pp.1–12.

Thio, N. & Karunasekera, S., 2005. Automatic measurement of a QoS metric for Web Service recommendation. In *Proceedings of the Australian Software Engineering Conference, ASWEC*. Brisbane, Australia: IEEE, pp. 202–211.

Tisue, S. & Wilensky, U., 2004. NetLogo: Design and Implementation of a Multi-Agent Modeling Environment. In *Proceedings of the Agent 2004 Conference on Social Dynamics*. Washington, D.C, pp. 1–20. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.294.8443.

TOGAF, 2011. A Glossary of Supplementary Definitions. *TOGAF*. Available at: http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap03.html.

TOGAF, 2004. *SLA Management Handbook*, The Open Group (TOGAF).

TOGAF, 2013. SOA Reference Architecture Technical Standard : Overview of the SOA RA Layers. *TOGAF*. Available at: http://www.opengroup.org/soa/source-book/soa_refar.

Tosi, D., Lavazza, L. & Morasca, S., 2015. Enriching Specifications to Represent Quality in Web Services in a Comprehensive Way. In *2015 IEEE Symposium on Service-Oriented System Engineering*. San Francisco Bay, USA: IEEE Computer Society, pp. 203–208. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7133530.

Tran, Q. et al., 2014. *Resource Management in Wireless Access Networks : A layer-based classification*, France: HAL - Inria.

Tupamäki, J. & Mikkonen, T., 2013. On the Transition from the Web to the Cloud. In *Web Systems Evolution (WSE), 2013 15th IEEE International Symposium*. Eindhoven, The Netherlands: IEEE, pp. 51–59.

Twomey, P. & Cadman, R., 2002. Agent-based modelling of customer behaviour in the telecoms and media markets. *Info*, 4(1), pp.56–63.

Udo, G.J., Bagchi, K.K. & Kirs, P.J., 2010. An assessment of customers' e-service quality perception, satisfaction and intention. *International Journal of Information Management*, 30(6), pp.481–492.

Upadhyaya, B. et al., 2015. Quality of experience: User's perception about web services. *IEEE Transactions on Services Computing*, 8(3), pp.410–421.

Upadhyaya, B. et al., 2014. Quality of experience: What end-users say about web services? In *2014 IEEE International Conference on Web Services, ICWS 2014*. Alaska, USA: IEEE, pp. 57–64.

Vaishnavi, V. and Kuechler, W., 2004. Design science Research in Information Systems. , p.desrist.org. Available at: http://desrist.org/desrist/article.aspx [Accessed November 28, 2014].

Van, W. et al., 2008. Confronting video-on-demand with television viewing practices. In *Innovating for and by Users*. COST and IBBT, pp. 1–222.

Venable, J.R., 2006. The role of theory and theorising in design science research. In *Proceedings of the 1st International Conference on Design Science in Information Systems and Technology*. Claremont, USA, pp. 1–18. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.2475&rep=rep1&type=pdf.

Vidgen, R. et al., 2011. Design Science Approach to Measure Productivity in Agile Software Development. In *European Design Science Symposium*. Verlag Berlin Heidelberg, pp. 171–177.

De Virgilio, R. & Torlone, R., 2009. A structured approach to data reverse engineering of Web applications. In *9th International Conference, ICWE 2009 San Sebastián, Spain, June 24-26, 2009 Proceedings*. Sebastian, Spain: Springer, pp. 91–105. Available at: http://dx.doi.org/10.1007/978-3-642-29066-4_11.

De Vleeschauwer, B. et al., 2008. End-to-end QoE optimization through overlay network deployment. In *2008 International Conference on Information Networking, ICOIN, IEEE*. Busan, Korea (South): IEEE, pp. 1–5.

Walls, J.G., Widmeyer, G.R. & El Sawy, O. a, 1992. Building an Infomiarion System Design Theory for Vigilant EIS. *Information Systems Research*, 3(1), pp.36–58.

Wamser, F. et al., 2013. Using buffered playtime for QoE-oriented resource management of YouTube video streaming. *European Transactions on Telecommunications*, 25(3), pp.11–20.

Wilensky, U., 1999. NetLogo. , p.http://ccl.northwestern.edu/netlogo/. Available at: https://ccl.northwestern.edu/netlogo/docs/copyright.html.

Windrum, P., Fagiolo, G. & Moneta, A., 2007. Empirical Validation of Agent-Based Models: Alternatives and Prospects. *Journal of Artificial Societies and Social Simulation*, 10(2).

Witten, I., Frank, E. & Hall, M., 2011. *Data Mining: Practical Machine Learning Tools and Techniques 3rd* Third., Morgan Kaufmann Publishers Inc. San Francisco, CA, USA ©2011. Available at: http://dl.acm.org/citation.cfm?id=1972514.

Wu, K., 2010. QoE-Based E2E Data Service Quality Improvement (SQI). , 11(1), pp.37–40.

Yalezo, S. & Thinyane, M., 2013. Architecting and Constructing an SOA Bridge for an MVC Platform. In *2013 Fourth World Congress on Software Engineering*. Hong Kong, China China: IEEE, pp. 45–49. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6754261.

Yamauchi, D. & Ito, Y., 2015. A Method of Evaluating Effect of QoS Degradation on Multidimensional QoE of Web Service with ISO-based Usability. *International Journal of Computer Networks & Communications(IJCNC)*, 7(1). Available at: http://arxiv.org/abs/1502.03612.

Yamauchi, D., Ito, Y. & Tahara, T., 2015. A Method of QoE Management in Online Shopping Web Services with TCP Variables. *INTERNETWORKING INDONESIA*, 7(1), pp.3–7.

Yamazaki, T. et al., 2012. A service quality coordination model bridging QoS and QoE.

In Quality of Service (IWQoS), 2012 IEEE 20th International Workshop on, pp.1–4. Available at: http://dl.acm.org/citation.cfm?id=2330748.2330779 [Accessed October 2, 2013].

Yetgin, Z. & Göçer, Z., 2015. Quality of experience prediction model for progressive downloading over mobile broadcast networks. *Telecommunication Systems*, 58(1), pp.55–66.

Yi, L. et al., 2012. A model for evaluating QoE of mobile Internet services. In *Proceedings of 2012 15th International Symposium on Wireless Personal Multimedia Communications (WPMC)*. Taipei, Taiwan: IEEE, pp. 438–442.

Zagorecki, A., Ko, K. & Comfort, L.K., 2010. Interorganizational Information Exchange and Efficiency : Organizational Performance in Emergency Environments. *Journal of Artificial Societies and Social Simulation*, 13(3), pp.1–11.

Zeng, L., Lei, H. & Chang, H., 2007. Monitoring the QoS for Web Services. In *Fifth International Conference –Service-Oriented Computing - ICSOC 2007*. Vienna, Austria: Springer, pp. 132–144. Available at: http://www.springerlink.com/index/n4770270k155l648.pdf.

Zhong, W. & Kim, Y., 2010. Using Model Replication to Improve the Reliability of Agent-Based Models. In *Advances in Social Computing, Proceedings*. Bethesda, MD, USA: Springer, pp. 118–127.

Zieliński, K. et al., 2012. Adaptive SOA solution stack. *IEEE Transactions on Services Computing*, 5(2), pp.149–163.

Zinner, T. et al., 2010. Summary to Impact of Frame Rate and Resolution on Objective QoE Metrics. In *In Quality of Multimedia Experience (QoMEX), 2010 Second International Workshop, IEEE*. Trondheim, Norway: IEEE, pp. 29–34.

Appendix

I. Extension of Literature Review

In order to effectively evaluate quality of Web Application and investigate whether it's intrinsic limitation is related to objective or subjective aspects or to both, however, it is important to understand the following characteristics: (1) Web Application Architecture; and (2) Quality of Web Application (Lew et al. 2012; Mich et al. 2003):

- **Web Application Architecture:** It is important to understand the architecture of Web Application (Lew et al. 2012) because it plays a vital role in determining quality attributes and sub-attributes of the components of the architecture (Babar 2008). According to Bass et al. (2001), most of quality attributes (e.g. performance, availability, reliability, accessibility, and success-ability) of software applications are generally derived from the design of software architecture, which includes a number of components, each of which somehow corresponds to a specific set of quality metrics.

- **Quality of Web Application:** In literature, there are three general approaches for evaluating quality of Web Application (Mich et al. 2003): (1) models for evaluating software quality such as ISO models (e.g. ISO 9126 for quality characteristics and ISO 14598 for process and guidelines); (2) models for usability-focused and human-computer interaction research (e.g. approaches which define quality in term of usability). (3) Models for website evaluation and design.

Web Application Architecture

The term architecture in software engineering is defined as "a fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution" (ISO-42010 2012 pp. 2) . Whilst, application architecture is defined as “A description of the structure and interaction of the applications as groups of capabilities that provide key business functions and manage the data assets” (TOGAF Document G116 2011 Part II Phace C). Application architecture in a web-context is represented as a client-server model that distinguishes client software (runs in a web browser) from server software (runs in the server-side). The communication between client and server revolves around the navigation of web pages, and the request and response messages (Conallen 1999).

Web Application is evolved from two-tier to N-tier to Service Oriented Architecture (SOA) and composite architectures, each of which is tied to a specific technology. The two-tier architecture includes two layers: client layer and server layer, the former handles the interface, while the latter handles the business logic and the database. The two-tier architecture fails to be a scalable architecture because any change on the business or database requires a new deployment. Consequently, the three-tier architecture is evolved with three layers: client layer (front-end component), business logic layer (middle-tier component), and database layer (back-end component). The purpose of adding the business logic layer is to handle the business logic of the application and control the interaction between the client and server layers. This enhances the scalability and reusability of the architecture. N-tier architecture is traditionally based on the three-tier architecture in which front-end, middleware, and back-end components are physically separated. Whilst, the N refers to the number of the logical layers between client and server. The level of abstraction in N-tier architecture enables the scalability and provides a loosely coupled architecture that can be efficiently extended and reused (Offutt 2002b).

With the 3-tier and N-tier approaches, Model View Controller (MVC) has been widely adopted and accepted as a standard design pattern for the development of web applications (Tupamäki & Mikkonen 2013) e.g. Oracle ADF and ASP.NET implement MVC to separate user interface, programming logic, and data layers from each other. MVC defines several interfaces that divide the application into three main components: model, view, and controller (Krasner & Pope 1988). A model stores an object that can be as simple as a string or an integer, or it can be a complex object such as table or query. The object is retrieved according to requests from the controller and displayed as a web page in the view. A view is a visual representation of the model (e.g. web page) that generates an output to the user. A controller manages the interfaces between models and views and inputs from user or system. Figure A-1 illustrates the interaction between the model, view, and controller components. Without the MVC model, it is hard to develop a modern web application that has a reusable and pluggable architecture, where components can be abstracted at a low level of abstraction, as well as designed in a way to isolate he business functions from each other.

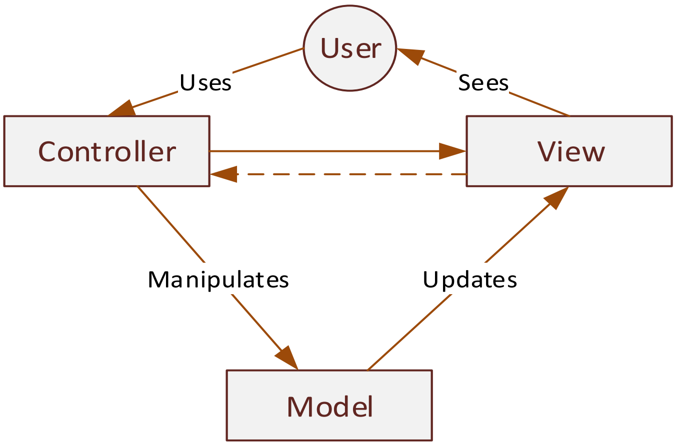


Figure A-1: Traditional MVC model

MVC and 3-tier approaches complement each other. MVC is an application and component design pattern that focuses on how the code is designed and presented to users, in which changes in one component do not required changes to another. While, 3-tier is an architectural pattern that focuses on how the interfaces of the application are separated and interacted. A fundamental concept in MVC and 3-tier architectures is the user never interacts directly with the data layer i.e. in the MVC, the interaction with the data is through the model component, while In the 3-tier, the interaction is made through the middle tier. When the MVC is integrated with the 3-tier architecture, the view and controller components in MVC are incorporated into the presentation layer in 3-tier, and the model component is incorporated into the business logic layer (Rawsthorne 2011). Figure A-2 illustrates how MVC is integrated with the 3 tier architecture.

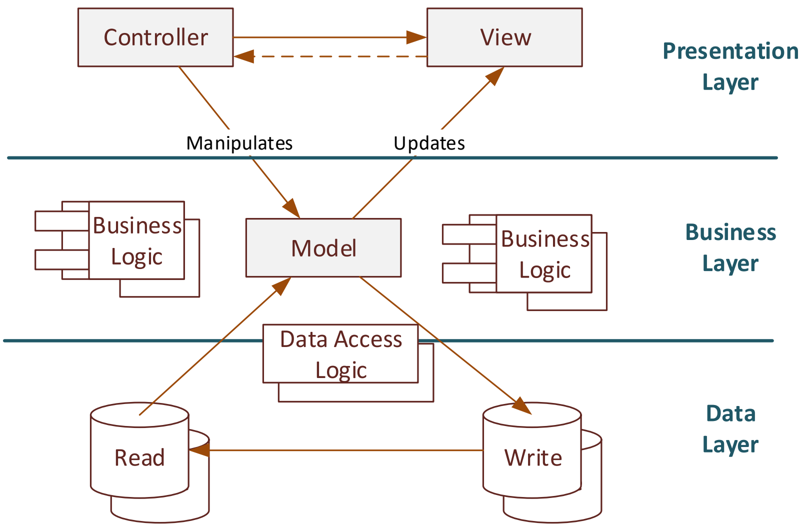


Figure A-2: MVC in three-tier architecture (Rawsthorne 2011)

For better modularisation, MVC has been mapped with SOA approach (Yalezo & Thinyane 2013), which promotes a high level of abstraction and interoperability between heterogeneous web applications and technologies. SOA is defined as "a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains" (OASIS 2005 pp. 8). The ultimate aim of SOA is to facilitate and manage the development of large-scale enterprise applications and internet-scale provisioning of services, and to reduce the cost of service integration and business collaboration. Fundamentally, SOA is introduced as an architectural pattern for describing loosely coupled software components; however it can be viewed from several different perspectives (Temnenco et al. 2010; IBM 2011). For instance, companies tend toward SOA approach to: (1) develop a standards-based enterprise service bus; (2) enable dynamic, extendable and flexible service mediation; (3) ensure effective governance and policy enforcement; and (4) promote and facilitate service reuse with control.

In the context of enterprise application architecture, TOGAF (2013) refined the SOA reference model (OASIS 2006) to facilitate a model-driven approach towards multi-tier architecture from a consumer and provider point of view with cross-cutting concerns presenting the architecture and its design principles. TOGAF defines a set of architecture building blocks that describe the SOA elements of service component architecture, as well as provides general rules and guidelines for developing n-tier SOA architecture as shown in Figure A-3, which includes nine layers, as follows:

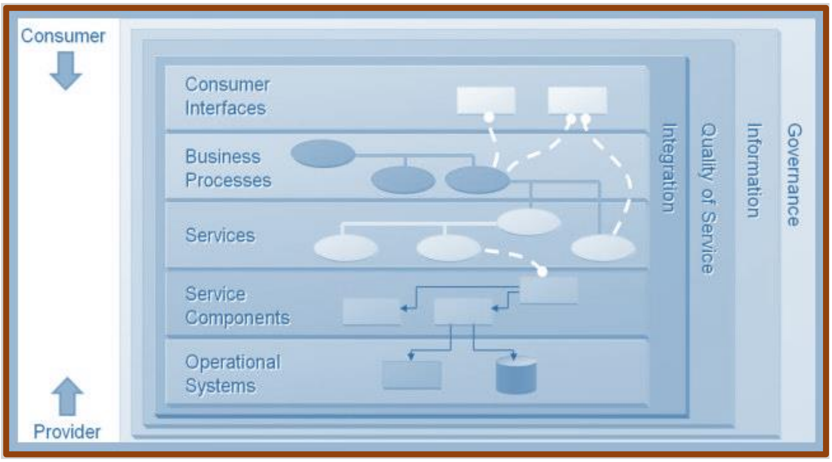


Figure A-3: Logical Solution View of the SOA RA (TOGAF 2013)

- **Operating Systems Layer:** encompasses the enterprise's portfolio of IT assets, including applications, middleware services, data, operating systems, infrastructure, and hardware.

- **Service Components layer:** consists of computer-based components that bind and implement services, where the concept of interoperability is desirable

- **Services Layer:** defines services which are offered by service provider and used by service consumer as abstract specifications of service components that invoke the business functions.

- **Business Processes Layer:** provides a number of composite services that orchestrate and maintain the information flow and interaction among a set of services and end-users

- **Consumer Interfaces Layer:** implements and handles the presentation layer, where a service consumer interacts with the provided services.
- **Integration Layer:** is a middleware layer that handles and manages the interactions across all system components and the interfaces between building blocks.
- **Quality of Service Layer:** monitors the quality of services (QoS) of the architected system (e.g. determines performance, availability, reliability, scalability, manageability, and security)
- **Information Layer:** includes building blocks that build information services capability i.e. information integration, management, security, business analytics, and information modelling.
- **Governance Layer:** ensures that the architecture adheres to the defined policies, guidelines, and standards which are applied in the organization as objectives, regulations, and strategies.

Keeping the focus of this research in mind, SOA can be adapted as an overarching approach as in Perrey and Lycett (2003) which defines the abstractions used in the development of the functional components of distributed systems. In this research, SOA is used to describe the abstraction layer between service, service provider, and service consumer, as well as to describe the abstractions of quality attributes across layers and along service provider and consumer. QoS layer in SOA helps to obtain a better understanding of the objective factors that influence quality of Web Application. In addition it has the ability to examine the overall quality of services requirements i.e. it has the ability to capture, monitor, and configure the quality attributes, which are generally derived from ISO 9126 standard (Franca & Soares 2015). There are many models that address quality attributes and metrics for web applications as discussed in the next section.

Quality of Web Application

Quality of Web Application is a set of characteristics and features that satisfy stakeholders' requirements (Losavio et al. 2003). Different stakeholders have different views on software quality i.e. end-user has a concern about quality of the final product. Software developer has a concern about quality of the development process, which produces the final product. Marketing manager has a concern about the marketing requirements. Hence, the overall quality of the software product can be determined by a combination of different views and concerns that are addressed during the software architecture process, which involves several stakeholders (Losavio et al. 2003). According to Franca and Soares (2015), quality of Web Application can be expressed as a multidimensional construct based on a variety of characteristics that are generally derived from ISO-9126 and ISO-25010 quality models, which are hierarchically structured (Behkamal et al. 2009). These quality characteristics and their sub-characteristics are as follows:

- **Functionality:** determines the capability of the software to provide the required system functions when it is used under specified condition. It includes the following factors: suitability, accuracy, interoperability, security, and compliance.

- **Reliability**: determines the capability of the software to maintain the required level of performance during its operational period and under various operational conditions. It includes the following factors: maturity, fault tolerance, recoverability, and compliance

- **Maintainability**: determines the capability of the software to perform modifications, changes, and customisation for implementing new requirements. It includes the following factors: analysability, changeability, stability, testability, compliance

- **Efficiency**: determines the capability of the software to optimise resources and enhance performance. It includes the following factors: time behaviour, resource behaviour, compliance
- **Portability**: determines the capability of the software to be adaptable and transferrable across different environments. It includes the following factors: adaptability, install-ability, co-existence, replace-ability, compliance
- **Usability**: It is defined as "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" (ISO 9241-11 1998). Usability is the capability of the software to be usable, attractive, and user friendly to the user, when there is an interaction between user and system for performing a specific task (Behkamal et al. 2009). It includes the following factors: understand-ability, Learnability, operability, compliance, attractiveness

The quality model is defined as "a set of characteristics and the relationships between them, which provide the basis for specifying quality requirements and evaluating the quality" (Behkamal et al. 2009). Each quality characteristic is divided into sub-characteristics, and each sub-characteristic is decomposed to a more concrete level, called quality attributes that are measured by metrics (Losavio et al. 2003). Attributes are divided into two types: Internal and external attributes (Franca & Soares 2015). Internal attributes are directly measured i.e. attributes that can be measured in terms of the software itself under specified condition. External attributes are indirectly measured i.e. attributes that are measured in terms of how the software interacts to its environment under specified condition such as reliability and maintainability (ISO/IEC 2008). Metrics are parameters or measures that express the state of the running software under certain condition through measurement methods and scales i.e. logical sequence of operations that measure software quality objectively (Lew et al. 2012).

In accordance with ISO-9126 and ISO-25010 quality models, there are many approaches presented for software quality assessment as it shown in the comparative study conducted by Kumar & Dadhich (2015), which evaluates the existing quality frameworks that are based on ISO quality models and proposed by contemporary researchers. The comparison is accomplished in terms of classification and sub-classification of quality characteristic among the existing web quality models. Generally, most of the proposed frameworks and quality models of Web Application provide guidelines and methodologies for modelling and classifying metrics, monitoring quality, measuring quality, and evaluating quality (Malak et al. 2004; Calero et al. 2004; Hasan et al. 2012).

For modelling and classifying web metrics, Ramler et al. (2002) presented a three-dimensional cube model that is commonly used. The model describes a generic schema including quality aspects, features, and phases to organise tests for quality aspects of Web Application. The classification on each dimension is principally based on the requirements of the software and its environment. The cube model is extended to consider stakeholders who are involved in the development, and ultimately it is applied by Ruiz et al. (2003) and Calero et al. (2005) to classify metrics and assess web quality. Calero et al. (2005) found that 44 percent of the classified metrics are extracted from the presentation layer, 48 percent are related to usability. In addition to these metrics, there are hundreds of other metrics that can be found in the literature but without guiding how they are measured. Therefore, other researchers followed the goal question metric approach, which is originally presented by Basili et al. (1994) to defined the goals behind measuring software quality, and to define the measures of these goals. For instance, Malak et al. (2004) presented a multidimensional model based on Ramler et al. (2002) that specifies the hierarchy of problems related to Web Applications quality using goal-oriented analysis and support tool for quality assessment and evaluation.

Building upon the above, Mich et al. (2003) developed a model that has seven dimensions identity, content, service, location, management, usability, and feasibility to helps webmasters and developers to generally evaluate the quality of

Web Application and incorporate the assessment finding into site design to examine how well the site meet requirements by examining the question of what-who-why-when-where-how. In the same sense, Signore (2005) developed a model that has five dimensions: correctness, presentation, content, navigation, and interaction to define Web Application measurement criteria, and to describe requirements and how they can be related to the quality characteristics. The process of evaluating the quality of Web Application is defined by Olsina & Rossi (2002), which includes four main technical phases: (1) specification of quality requirements phase; (2) elementary evaluation phase (during design and implementation); (3) global evaluation phase; and (4) conclusion and recommendations. The process considers the Web Application characteristics and attributes from the user's viewpoint rather than software attributes such as design or code quality.

Monitoring quality of Web Application has become a vital and essential process in many enterprises (Hasan et al. 2012). It is generally performed as a part of SLA management (Thio & Karunasekera 2005) thus service provider is required to maintain the agreed service level regularly and efficiently. To achieve monitoring systematically and dynamically on service operations, the metrics which are defined by ISO and classified by Ramler et al. (2002) and other researchers need to be fundamentally classified in terms of client-server approach as it is addressed in Zeng et al. (2007). Consequently, QoS metrics are classified into three groups: (1) provider-advertised metrics that are provided by service provider; (2) consume-rated metrics that are measured based on consumers' feedback and evaluation; and (3) observable metrics, which are based on monitored service operational events. According to Hasan et al. (2012), QoS of web application can be monitored in two ways: (1) server-side, which provides more accurate and real-time monitoring results; and (2) client-side, which provides less accurate results due to the influence by other factors during the interaction with service provider. Focusing on the interaction of consumer and provider, SOA approach has the capability for managing and monitoring the software services and applications. In addition SOA provides a mechanism to monitor and enforce governance and policies and corresponding business rules across system components, including security policies, business-level policies, data access, and system privileges (TOGAF 2013).

Measuring and monitoring quality of Web Application are interdependent processes that complement each other. In monitoring process, metrics, their acceptance value, and actions to be taken are defined, while in measuring process, formulas, scales, and weight of calculating the obtained metrics are formulated as it is expressed in ISO/IEC TR 9126-3 (2002) and OASIS (2012). In another word, measuring process itself is a function of monitoring process. OASIS (2012) introduced a service level measurement quality model for measuring the quality which is perceived by users when the web service is actually used with respect to the scope of SOA. Hence OASIS (2012) model does not consider all the quality factors and sub-factors which are defined by ISO-9126 and ISO-25010 (Tosi et al. 2015). OASIS essentially focuses on SOAP-based web service (Simple Object Access Protocol), which is used as a protocol specification for exchanging messages in the implemented web service. Consequently OASIS (2012) formulated general equations and methods for measuring the following quality factors:

- **Performance:** measures how fast a service provider responds to the requested service. It can be described by sub-factors such as response time and maximum throughout, which are measured by millisecond and calculated by the ratio of the maximum number of completed requests to unit time.

- **Availability:** computes the ratio of time period in which a service is available and ready to use i.e. by assuming that up-time is the time when a service is available and down-time is the time when a service is not available. Availability is measured by a percentage (%) of availability over a time period and calculated by the following formula, where unit-time is a unit to measure the time

- **Reliability:** measures the ability of a Web Application to achieve its required jobs under a specific set of conditions at a particular time interval i.e. it indicates the number of service failures over a period of time. Reliability is generally computed as a function of the following characteristic: Accuracy (C), fault tolerance (F), testability (T), interoperability (I), availability (A), and performance (P) (Susila & Vadivel 2014). Reliability = f(Cc, Ff, Tt, Ii, Aa, Pp) where c, f, t, i, a, and p are the weight associated with the occurrence of each characteristic.

- **Success-ability:** indicates the degree to which a web service is accomplished in a specific time according to an agreed time e.g. SLA. It measures the ratio of the number of successful service responses to the number of service requests i.e. it measures the percentage of successful messages.

- **Accessibility:** represents the degree to which a web service is accessible. For example, in certain circumstances, some resources need to be accessible for end users, even if the service is not available (OASIS 2012). Accessibility is normally tested by returning an acknowledgement message for a request message, thus it can be calculated by the ratio number of acknowledgement messages to the number of request messages.

Obviously, there is relatively limited guidance to measure the level of usability of web service. Most of the quality models based on ISO-9126 (including ISO-9126) do not consider the relation between phases in the software development cycle, usability, and user experience. In addition they defined the usability metrics in a high abstraction level, hence usability is often measured without a consistent method and consolidated framework (Seffah et al. 2006). Despite this limitation, several recent studies have been presented examining the relationship between usability and user experience. For instance, Yamauchi & Ito (2015) applied ISO 9241-11 as multidimensional quality model for web service, and Bevan et al. (2015) evaluated ISO 9241-11 model and identify the new measurements needed for usability measurement. Mifsud (2011) clarified the difference between usability and user experience i.e. the central point of usability is the goal achievement when interacting with a Web Application, while user experience is a result of usability.

Seffah et al. (2006) and Mifsud (2015) presented a generic framework based on ISO 9241 to quantify and measure the usability of Web Applications, considering the following characterises:

- **Effectiveness:** implies the success of achieving goals. It focuses on the service accuracy and completeness, and can be measured by calculating the ratio of the mean difference between the number of tasks completed successfully and the total number of tasks to be achieved (Bevan et al. 2015) i.e. it evaluates how long it takes a user to accomplish a tasks in a unit time interval

- **Efficiency:** implies the success of achieving goals without wasting time (Bevan et al. 2015) i.e. It evaluates how efficiency a task is accomplished by a normal user (in a unit time interval), comparing with an advanced user on the same service (Seffah et al. 2006).

- **Satisfaction:** implies the willingness of using the system (Bevan et al. 2015). It refers to the response, which is subjectively determined from users, expressing their feelings and experiences about the services they receive. In term of usability, there are several approaches that can be used to measure satisfaction (Madhavan & Alagarsamy 2014) but the most commonly used are: (1) task level satisfaction — after users attempt a task, they are given a post-task questionnaires to express how difficult the given task was. Typically, the questionnaire is designed with Likert-scale ratings form (Franca & Soares 2015) that has up to five questions to be answered by users (Mifsud 2015); (2) test level satisfaction — at the end of the test session, users are asked to answer few questions regarding their impression of the overall ease-of-use (Mifsud 2015).

- **Learnability:** represents the degree to which the web service enables users to understand and learn the provided functions i.e. it enables users to productively use the service by performing a particular action within minimal time (Seffah et al. 2006). It can be measured by computing the ratio between the time taken to perform a particular task and the average time of performing the same task by other users

- **Operability:** indicates the degree to which an application or service has the facilities to make system easy to operate, maintain and control (Franca & Soares 2015). It can be measured by computing the ratio between the number of instances of operations with consistent behaviour and the total number of operations.

The above usability characteristics and sub-characteristics can be expressed as quality factors that are closely aligned to the Graphical User Interface (GUI) components as they are directly dependent on user's interaction, and independent from the software development cycle (Losavio et al. 2003). Therefore, the revised version of ISO 9241-11 (1998) highlighted the relationship between usability and some other associated approaches such as user-centred quality of service and user experience. This allows users to be actively involved and participate in the usability assessment process, i.e. they evaluate the design and its usability from their perspective (Bevan et al. 2015).

On the other hand, in contrast to the above quality models which are fundamentally related to technical and objective aspects, some researchers argue that quality of Web Application can be affected by non-technical aspects (Negash et al. 2003; Udo et al. 2010) e.g. governance, customer services and IT support. According to the assessment conducted by Udo et al. (2010) regarding to customer's e-services quality perception, there are five non-technical factors that can affect quality of web services including:

- **Tangible**: is the appearance of the physical personal, communication material, facilities, and equipment;

- **Ability**: is the capability to perform the provided service accurately as promised;

- **Responsiveness**: is the willingness to support users' needs efficiently and promptly;
- **Assurance**: is the professionalism knowledge of support representatives to help users confidently and solve their problems with high quality solutions;
- **Empathy**: is the ability to consider the emotional experience and impact of users. In addition to these main factors, Negash et al. (2003) extended sub-factors to assess services quality in term of effectiveness towards non-technical aspects.

II. Screenshots of Artefacts

Iteration 1

This section provides screenshots of the Web Service metrics, which are taken from Oracle WebLogic console. Figure A-4 shows the values related to the performance of Web Service. The values of the metrics are generated and stored into a log file to be interpreted, summarised, and analysed using ad-hoc techniques.

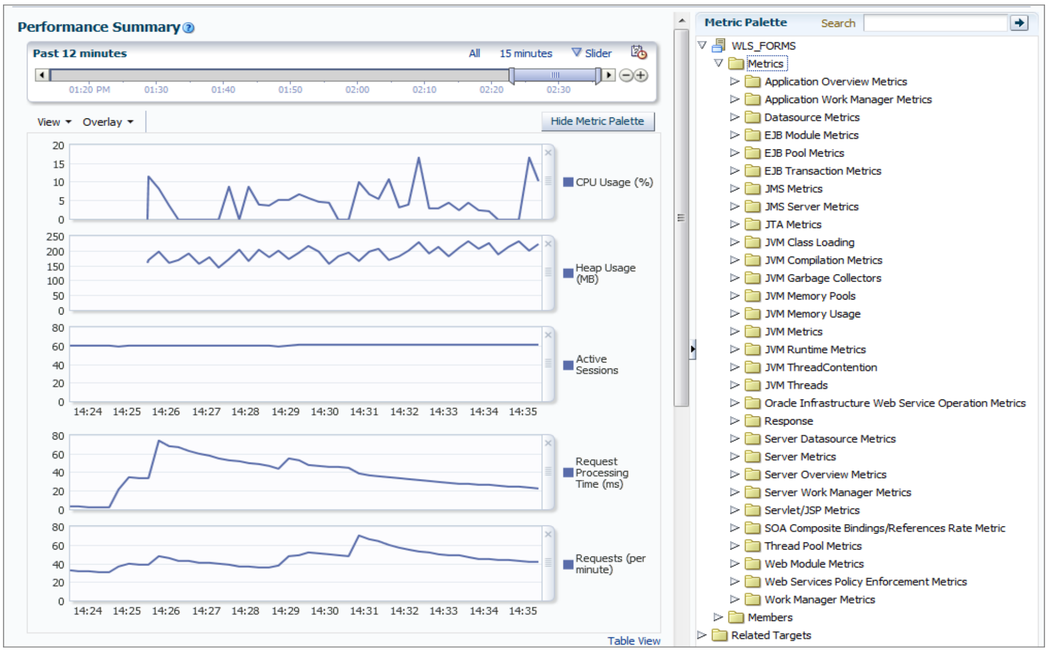


Figure A-4: Performance Summary

Figure A-5 provides a screenshot of requests processed by the Web Server, showing the module, application, average and total clients processing time (per millisecond),

Most Requested
Servlets and JSPs

Name	Web Module	Application	Requests Processed	Average Client Processing Time (ms)	Requests (per minute)	Total Client Processing Time (ms)
FileServlet	/inspection.wsil	wsil-wls	0	0	0.00	0
frmservlet	forms	formsapp(11.1.2)	289	4	0.00	1,038
JspServlet	dms.war	DMS Application(11.1	0	0	0.00	0
JspServlet	forms	formsapp(11.1.2)	0	0	0.00	0
JspServlet	/inspection.wsil	wsil-wls	0	0	0.00	0
JspServlet	/formsconfigmbeans	formsconfigmbeans	0	0	0.00	0

Figure A-5: Requests Processed

Figure A-6 provides a screenshot of the user session form, which provides information about process ID, IP address, configuration session, memory and CPU usage, and database name.

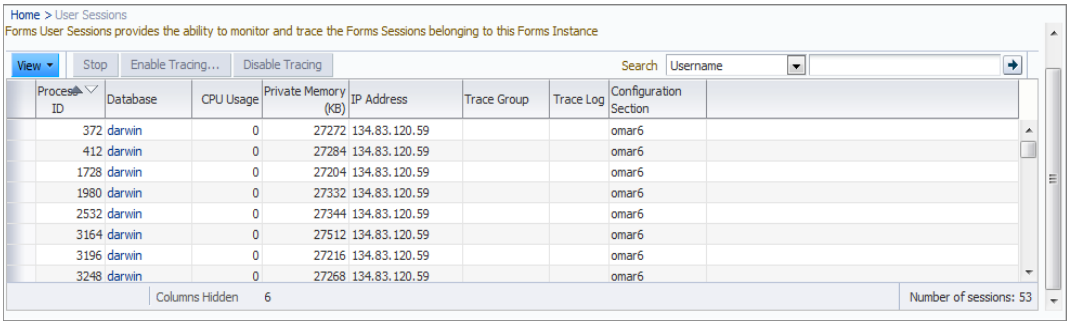


Figure A-6: User Sessions

Figure A-7 provides a screenshot of the form which generates the error messages, including incident error, warning, notification, trace, and unknown messages. The data which is extracted and analysed from this log is used for measuring the reliability of the web application.

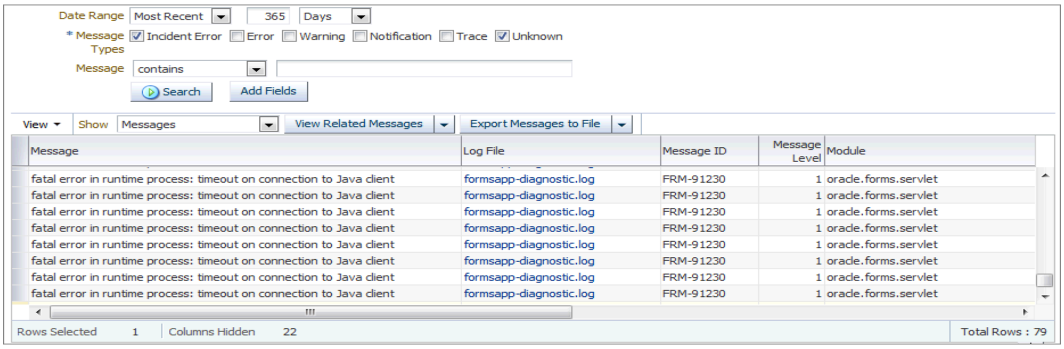


Figure A-7: Log Error Messages

Iteration 2

This section provide screenshots of WEKA interfaces. Figure A-8 provides a screenshot of Weka explore, showing the independent variable (feature) and the dependent features (target)

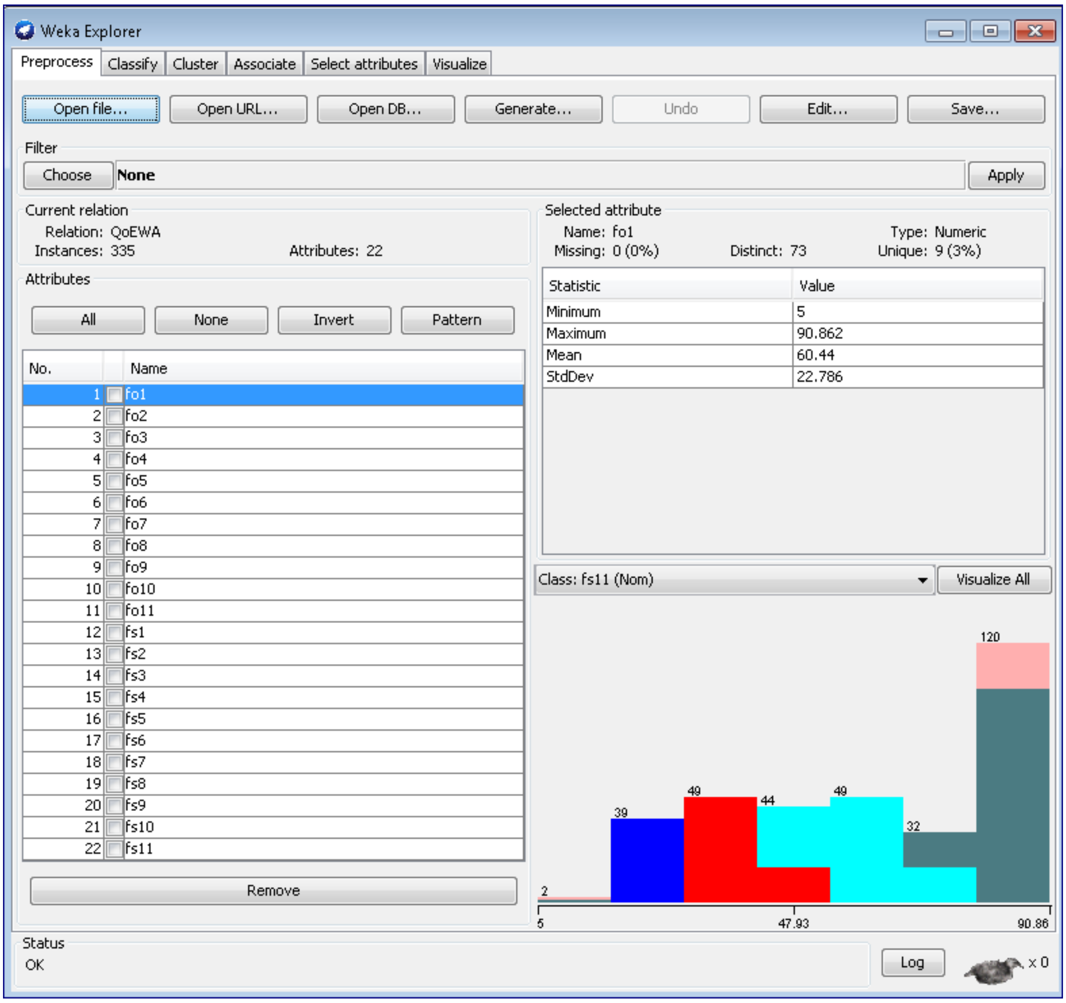


Figure A-8: WEKA explorer showing the attributes of the dataset

Figure A-9 provides a screenshot that shows the details of the ARFF file, which is used to feed WEKA with the training dataset. The features are defined as numbers, while the target are defined as a list of values describing the classification.

QoEWA - Copy.arff - Notepad

File Edit Format View Help

```
@relation 'QoEWA'
@attribute fo1 numeric
@attribute fo2 numeric
@attribute fo3 numeric
@attribute fo4 numeric
@attribute fo5 numeric
@attribute fo6 numeric
@attribute fo7 numeric
@attribute fo8 numeric
@attribute fo9 numeric
@attribute fo10 numeric
@attribute fo11 numeric
@attribute fs1 {Poor,Bad,Fair,Good,Excellent}
@attribute fs2 {Poor,Bad,Fair,Good,Excellent}
@attribute fs3 {Poor,Bad,Fair,Good,Excellent}
@attribute fs4 {Poor,Bad,Fair,Good,Excellent}
@attribute fs5 {Bad,Poor,Fair,Good,Excellent}
@attribute fs6 {Poor,Bad,Fair,Good,Excellent}
@attribute fs7 {Poor,Bad,Fair,Good,Excellent}
@attribute fs8 {Bad,Poor,Fair,Good,Excellent}
@attribute fs9 {Bad,Poor,Fair,Good,Excellent}
@attribute fs10 {Poor,Bad,Fair,Good,Excellent}
@attribute fs11 {Bad,Poor,Fair,Good,Excellent}

@data

28,27,27,27,26,26,26,25,25,26,25,Poor,Poor,Poor,Poor,Bad,Poor,Poor,Bad,Bad,Poor,Bad
34,33,33,33,32,31,31,30,31,32,31,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
21,20,20,20,20,19,19,18,19,20,19,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad
33,32,32,32,31,31,30,29,30,31,30,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
34,33,33,33,32,31,31,30,31,32,31,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
36,35,35,35,34,33,33,32,32,34,32,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
31,30,30,30,29,29,28,27,28,29,28,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
20,19,19,19,19,19,18,18,18,19,18,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad
25,24,24,24,24,23,23,22,22,23,23,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad
38,37,36,37,36,35,35,33,34,35,34,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
34,33,33,33,32,31,31,30,31,32,31,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
30,29,29,29,28,28,28,26,27,28,27,Poor,Poor,Poor,Poor,Bad,Poor,Poor,Bad,Bad,Poor,Bad
30,29,29,29,28,28,28,26,27,28,27,Poor,Poor,Poor,Poor,Bad,Poor,Poor,Bad,Bad,Poor,Bad
20,19,19,19,19,19,18,18,18,19,18,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad
40,39,38,39,38,37,37,35,36,37,36,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
22,21,21,21,21,20,20,19,20,21,20,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Bad
40,39,38,39,38,37,37,35,36,37,36,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor,Poor
27,26,26,26,25,25,25,24,24,25,24,Poor,Poor,Bad,Bad,Bad,Bad,Bad,Bad,Bad,Poor,Bad
```

Figure A-9: ARFF file

Iteration 3

This section provides screenshots of the code written to build the simulator. Figure A-10 shows the editor of NetLogo and the code that is used to optimise QoE.

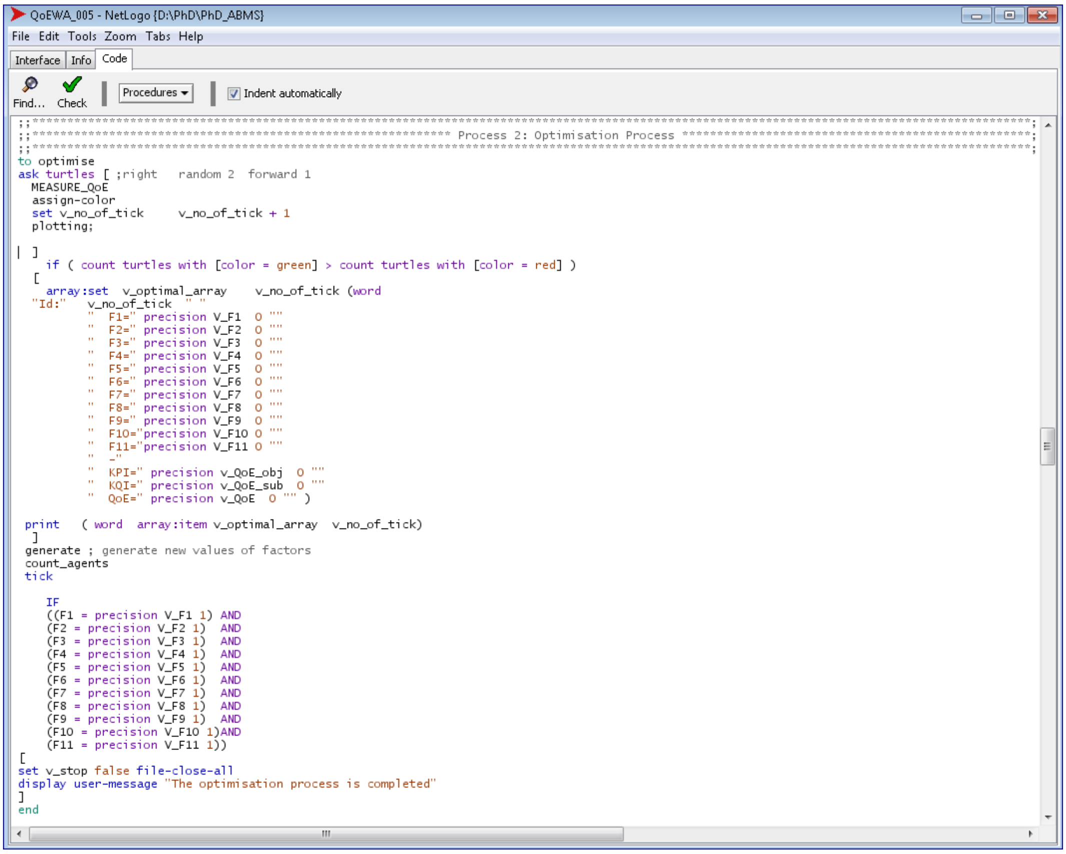


Figure A-10: Optimisation Process

Iteration 4

This appendix provides screenshot of the NetLogo code, which is used for developing the ABM system. Figure A-11 provides a screenshot of the rules that are defined (in Iteration 2) to enable QoEWA to predict the KPIs vales.

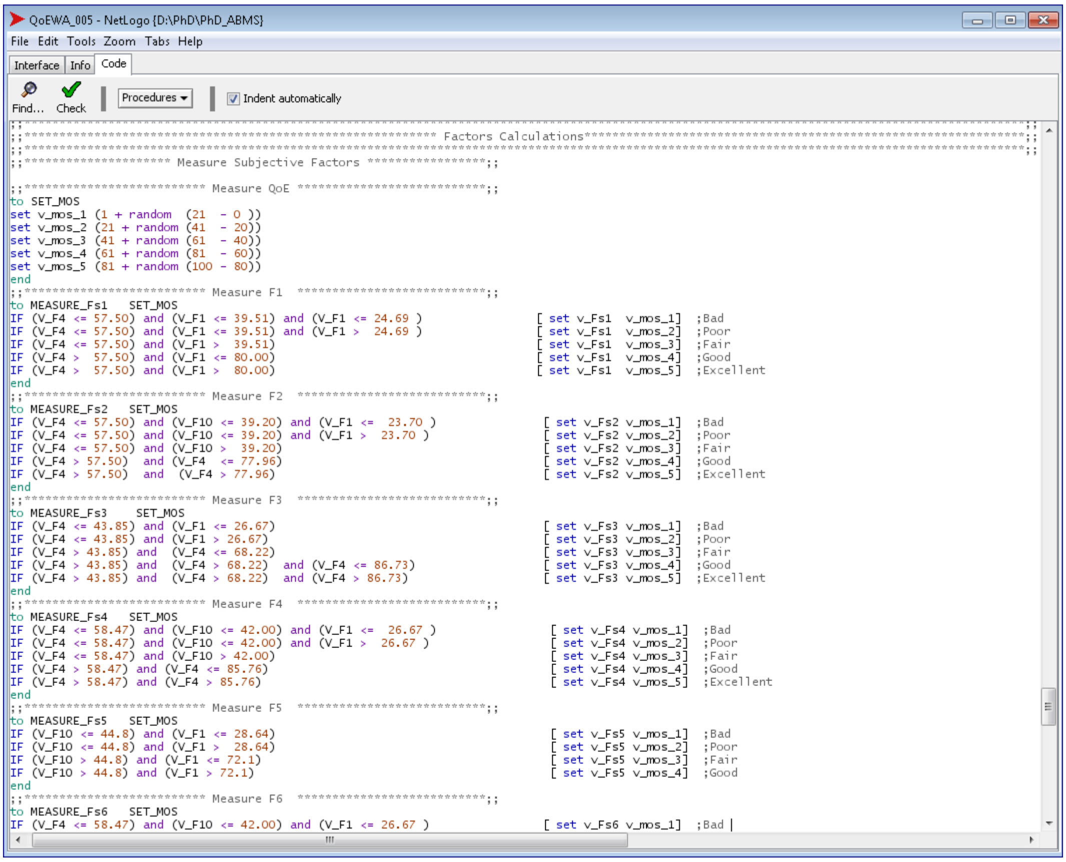


Figure A-11: The predictive rules for measuring KQI

Figure A-12 shows the code that is used for importing dataset into the ABM system. This is processed by the exploration process.

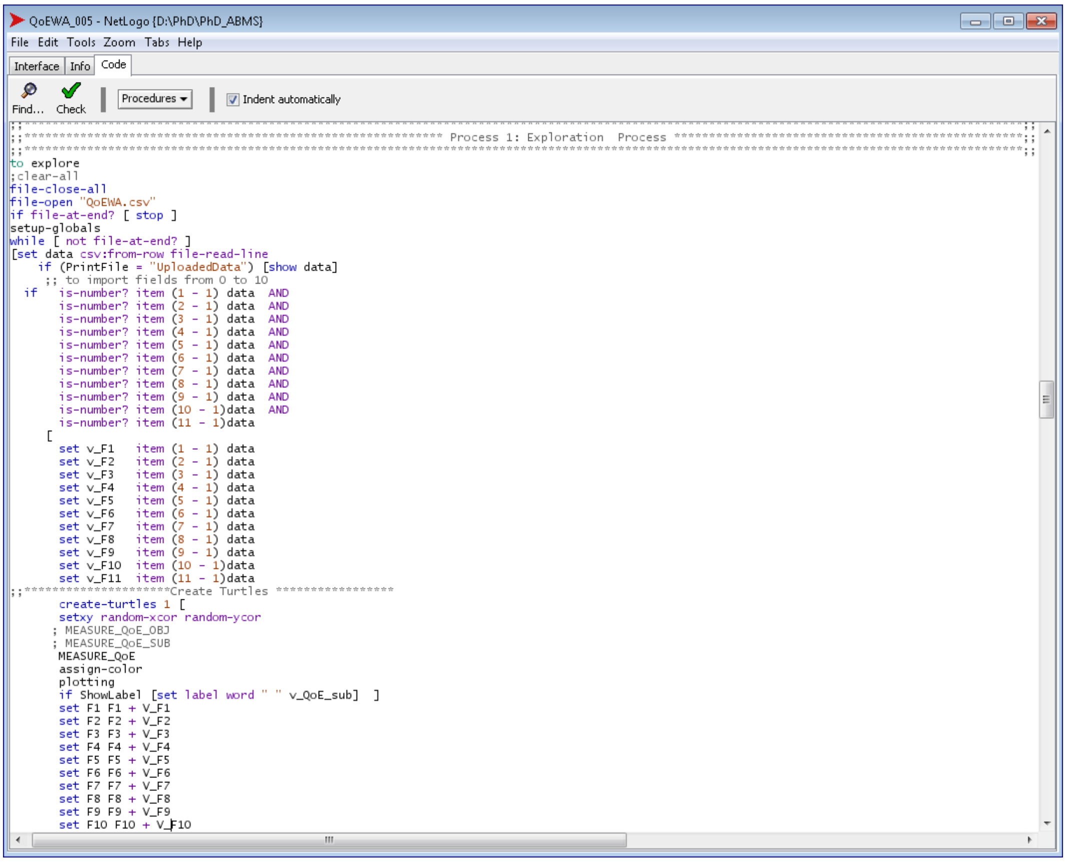
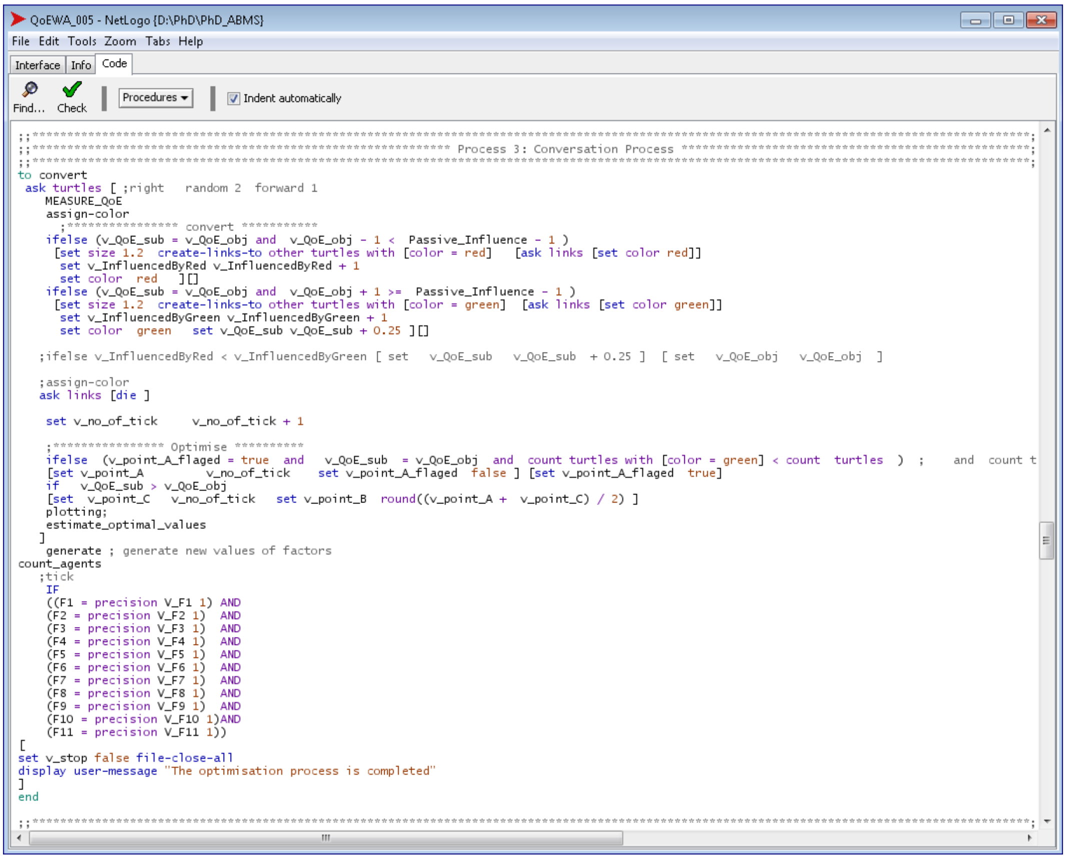


Figure A-12: The code of the exploration process

Figure A-13 shows the code examines the status of promotors and detractors agents, as well as determines the number of promotors and detractors converted from being passives.




```
;;**************************************************************************************************************************;
;;************************************************************ Process 3: Conversation Process *****************************;
;;**************************************************************************************************************************;
to convert
 ask turtles [ ;right   random 2  forward 1
   MEASURE_QoE
   assign-color
     ;**************** convert ***********
   ifelse (v_QoE_sub = v_QoE_obj and  v_QoE_obj - 1 <  Passive_Influence - 1 )
    [set size 1.2  create-links-to other turtles with [color = red]   [ask links [set color red]]
     set v_InfluencedByRed v_InfluencedByRed + 1
     set color  red   ][]
   ifelse (v_QoE_sub = v_QoE_obj and  v_QoE_obj + 1 >=  Passive_Influence - 1 )
    [set size 1.2  create-links-to other turtles with [color = green]  [ask links [set color green]]
     set v_InfluencedByGreen v_InfluencedByGreen + 1
     set color  green   set v_QoE_sub v_QoE_sub + 0.25 ][]

  ;ifelse v_InfluencedByRed < v_InfluencedByGreen [ set   v_QoE_sub   v_QoE_sub  + 0.25 ]  [ set   v_QoE_obj   v_QoE_obj  ]

  ;assign-color
  ask links [die ]

   set v_no_of_tick     v_no_of_tick + 1

   ;**************** Optimise **********
   ifelse  (v_point_A_flaged = true  and   v_QoE_sub  = v_QoE_obj  and  count turtles with [color = green] < count  turtles  )  ;    and  count t
   [set v_point_A         v_no_of_tick    set v_point_A_flaged  false ] [set v_point_A_flaged  true]
   if   v_QoE_sub > v_QoE_obj
   [set  v_point_C   v_no_of_tick   set v_point_B  round((v_point_A +  v_point_C) / 2) ]
   plotting;
   estimate_optimal_values
  ]
   generate ; generate new values of factors
count_agents
  ;tick
   IF
   ((F1 = precision V_F1 1) AND
   (F2 = precision V_F2 1)  AND
   (F3 = precision V_F3 1)  AND
   (F4 = precision V_F4 1)  AND
   (F5 = precision V_F5 1)  AND
   (F6 = precision V_F6 1)  AND
   (F7 = precision V_F7 1)  AND
   (F8 = precision V_F8 1)  AND
   (F9 = precision V_F9 1)  AND
   (F10 = precision V_F10 1)AND
   (F11 = precision V_F11 1))
[
set v_stop false file-close-all
display user-message "The optimisation process is completed"
]
end

;;**************************************************************************************************************************;
```

Figure A-13: The code of the interaction process